


Article

# Collecting Sensed Data with Opportunistic Networks: The Case of Contact Information Overhead

Tekenate E. Amah <sup>1,\*</sup>, Maznah Kamat <sup>1</sup> , Kamalrulnizam Abu Bakar <sup>1</sup>,  
Syed Othmawi Abd Rahman <sup>1</sup>, Muhammad Hafiz Mohammed <sup>1</sup>, Aliyu M. Abali <sup>1</sup>,  
Waldir Moreira <sup>2,3</sup> and Antonio Oliveira Jr. <sup>4</sup>

<sup>1</sup> Department of Computer Science, Faculty of Computing, Universiti Teknologi Malaysia (UTM), 81310 Skudai, Johor, Malaysia; kmaznah@utm.my (M.K.); knizam@utm.my (K.A.B.); syedothmawi@utm.my (S.O.A.R.); mhafizm@utm.my (M.H.M.); aliyuabali@gmail.com (A.M.A.)

<sup>2</sup> Fraunhofer-AICOS, Porto 4200-135, Portugal; waldir.junior@fraunhofer.pt

<sup>3</sup> PPGMO, Federal University of Goiás, Catalão 75704-020, GO, Brazil

<sup>4</sup> INF, Federal University of Goiás, Goiânia 74690-900, GO, Brazil; antonio@inf.ufg.br

\* Correspondence: amatek008@gmail.com; Tel.: +6-016-235-0389

Received: 27 July 2017; Accepted: 1 September 2017; Published: 5 September 2017

**Abstract:** The rising human population in urban environments drives the mission towards smart cities, which envisions a wide deployment of sensors in order to improve the quality of living. In this regard, opportunistic networks (OppNets) present an economical means of collecting delay tolerant data from sensors to their respective gateways for providing various Smart City services. Due to the distributed nature of the network, encounter-based routing protocols achieve acceptable throughput by requiring nodes to exchange and update contact information on an encounter basis. Unfortunately, sufficient insight into the associated overhead is lacking in the literature. Hence, we contribute by modelling contact information overhead and investigating its impact on OppNet routing, particularly in terms of data exchange success and energy consumption on portable handheld devices. Our findings reveal that the expected contact information overhead in Smart City scenarios significantly reduces data exchange success and increases energy consumption on portable handheld devices, thereby threatening the feasibility of the technology. We address this issue by proposing an algorithm that can be incorporated into encounter-based routing protocols to reduce contact information overhead without compromising throughput. Simulation results show that our proposed algorithm reduces the average contact information overhead, increases throughput and reduces average energy consumption.

**Keywords:** opportunistic networks; delay tolerant data; sensed data collection; routing; wireless sensors; Smart City; Internet of things; smartphones

## 1. Introduction

Opportunistic networks (OppNets) are based on the store-carry-forward (SCF) communication paradigm, in which data-bundles (or messages) are stored in device memory, physically carried from one point to another as the device moves about, and forwarded through available wireless communication interfaces (e.g., Bluetooth or Wi-Fi) when devices encounter each other (i.e., are within radio transmission range). This way, a message generated at a source can be delivered to its destination through multiple hops. OppNets therefore provide economical, infrastructure-less and mobile communication that involves delay-tolerant information. OppNets have been envisioned for advertisements in the form of digital word-of-mouth services, recommendation and opportunistic trading [1–3], as well as communication in the aftermath of large-scale disasters [4–6]. The following section presents another envisioned application of OppNets towards realizing smart cities.

### 1.1. Sensed Data Collection with OppNets

Human population is growing rapidly—at an average rate of 1.2% per annum over the last 50 years—and increasingly flowing towards urban areas [7]. This necessitates the cause towards smart sustainable cities, where there are preventive maintenance activities and resource optimization for critical infrastructure such as transportation, communications, water, and energy. In this regard, information communications technology (ICT) is tasked with developing economical and pervasive solutions to improve the quality of life through efficient urban operations and services. Wireless sensors play an important role in the realization of this goal. With the concept of Internet of things (IoT), which would see sensors widely deployed in the environment and embedded in physical objects, the applications are almost limitless.

The success of IoT relies on connecting these sensors to the Internet in order to share the generated information across multiple platforms and applications. This brings about urban automation networks (UANs), in which a backhaul is required to collect and transmit sensed data to dedicated gateways that are connected to remote management centers (through the Internet) where the information can be processed and action can be taken accordingly [8]. In UANs, the desired level of coverage can be directly achieved through long-range communication technologies or by deploying numerous relay nodes. However, it is still challenging to achieve the required level of scalability (w.r.t. the cost of procuring, installing, and maintaining supporting infrastructure) for every application with finite resources and limited budgets [9]. Equipping each sensor node with cellular connectivity is not economical (e.g., Du et al. [10] report an annual cost of \$4550 for 12 sensors) and also reduces sensor lifetime due to high transmission power. Setting up wireless mesh networks with ad-hoc technologies, which requires relay nodes to collect data from sensors, may not be economical as well (e.g., 1096 relays were required to collect CO<sub>2</sub> data from only 100 sensor nodes spread across Wuxi City, China [11]). Existing communication infrastructure in cities (e.g., networks offering Internet access to citizens) may also serve as the backhaul [8]. However, amidst the rapid growth of mobile data traffic due to the widespread use of mobile devices and content-centric services such as live audio and video streaming among mobile users [12], introducing data generated from sensors raises further difficulties in maintaining the quality of service.

A subset of Smart City applications tolerate infrequent sensor node connectivity opportunities (e.g., twice per day) and deal with data that can afford delays of up to hours or a day (e.g., agricultural monitoring, [13] habitat monitoring [14] and environmental monitoring data for: garbage collection and green zone management [8]; analysis of noise levels and water quality [15]; river pollution management [16]; and for meter readings [17]). In this regard, OppNets present a logical backhaul solution by leveraging pervasive mobile devices as data mules in a scalable manner to reduce costs in procuring, installing, and maintaining supporting infrastructure, e.g., [18–20]. At the same time, OppNets serve as a complementary technology for offloading cellular networks and freeing bandwidth for mobile users through direct device-to-device communication using available short-range wireless communication interfaces, e.g., [21–23].

Inspired by these benefits, researchers have studied the collection of sensed delay-tolerant data with OppNets from various perspectives. For example, Aguilar et al. [24] provide an analytical and experimental study of the performance and trade-offs of Bluetooth Low Energy as a wireless technology for collecting sensed delay-tolerant data. Can and Demirbas [19] evaluate the feasibility of collecting sensor data with OppNets by analyzing a city-wide mobility dataset of Beijing. Their spatial analysis suggest that knowledge of daily travel patterns of users may contribute in designing more effective data collection protocols. Wu et al. [25] identify significant patterns in the mobility traces of smartphone users that can be exploited to realize protocols for opportunistically conveying delay-tolerant data from sensors to their corresponding servers. The authors also identify important characteristics of user mobility—such as strong spatial and temporal localities—that need to be considered when designing related protocols and algorithms. Shi et al. [20] implement an opportunistic network testbed for a large-scale Smart City platform (i.e., platforms used to analyze data and extract valuable information

for city management and control), in order to evaluate solutions proposed for sensed delay-tolerant data collection with mobile phones. The authors propose a middleware that collects data from sensing infrastructure and sends them to the Smart City platform through multi-hop opportunistic routing. Regarding extending the lifetime of sensor nodes through duty-cycling, Wu et al. [26] propose a sensor node-initiated probing mechanism by exploiting rush hours, during which encounters with portable handheld user devices occur more frequently.

### 1.2. Contact Information Overhead in OppNets

The pervasiveness of portable handheld user devices (e.g., smartphones and tablets) makes them inseparable components of OppNets. These devices are able to serve as relay nodes and can cover areas of the city where other mobile nodes such as vehicles may be unable to reach. Unlike other mobile nodes, portable handheld user devices can readily participate in OppNets as they are already endowed with the necessary enablers such as short-range wireless communication interfaces (e.g., Bluetooth and Wi-Fi), computational capability and memory. They also present the primary user interface, as they are the main platform for human communication today. These devices can now be perceived as the users themselves: their attachment to humans allows them to follow and learn user movement patterns, gather social information about the user, as well as maintain user contextual information (e.g., contact information, occupation and preferences). They may even go as far as reflecting user emotion, as resource utilization on these devices could determine user satisfaction and willingness to participate in the network [27,28]. Hence, accounting for them in OppNets is a necessity.

Portable handheld user devices are resource constrained especially in terms of energy. In order to collect sensed data, they are required to interact with a wide range of nodes (e.g., sensors, smart vehicles, appliances and supporting infrastructure), and as per the message forwarding approach of most applicable OppNet protocols, contact information (i.e., encounter-based information) may need to be maintained, exchanged and updated each time nodes encounter each other. As human population—especially in urban areas—rises rapidly [7], so is expected of the nodes that participate in OppNets. Considering bandwidth limitations and the multi-purpose nature of portable handheld devices, it is desirable that the toll OppNets take on available resources remains minimal, without the tendency to rise at a high rate with user and device population.

Unfortunately, the current state of OppNet research shows that this is not the case. Emerging encounter-based protocols continue to adopt the same message forwarding approach: to compute nodes' ability to contribute to message delivery, they require information to be maintained for (at least) each encountered node, exchanged and updated during each encounter. Besides the large number of nodes expected in urban environments, there is also the tendency of high population density and dynamicity, which results in high encounter rates and frequent disconnections—imagine the number of encounters and the rate of disconnections a handheld device would experience in a shopping mall, train station, stadium or on the road during rush hours. Hence, the overhead associated with maintaining, exchanging and updating contact information tends to be significant. Unfortunately, the impact of this overhead remains unclear up to date, as it is often overlooked in OppNet protocol evaluations—for instance, Wang et al. [29] identify this overhead without studying its impact on available resources and network performance. Therefore, this paper investigates the impact of this overhead on: (i) data exchange success (i.e., the amount of data messages in node buffers that can be sent and received within the encounter duration), since encounters are often short-lived; and (ii) energy consumption on portable handheld devices.

### 1.3. Authors' Contribution

To avoid confusing this overhead with others used in OppNet evaluation, we employ the term “contact information overhead” which we define as the overhead incurred in maintaining, exchanging and updating contact information. In this regard, the following are questions yet to be answered,

which will give insight on the feasibility of existing routing approaches and take them a step closer to real-world implementation:

1. How can contact information overhead be modelled and what is its impact on data exchange success and the energy consumption on portable handheld devices?
2. How can contact information overhead be minimized without compromising throughput?

We address these questions in the following sections and contribute by: (i) modelling contact information overhead and investigating its impact on data exchange success and energy consumption on portable handheld devices; and (ii) proposing a forwarding algorithm namely Point-of-interest Forwarding (PoiFord) that can be incorporated into existing encounter-based routing protocols to reduce contact information overhead without compromising throughput.

#### 1.4. Organization of the Paper

The remainder of this paper is organized as follows. Section 2 provides the problem background and the need to address the existing issues in the following sections. In particular, we describe how portable handheld devices incur contact information overhead in the process of exchanging summary vectors, the lack of insight regarding this and our contribution in this regard. Then we present a brief overview of alternative forwarding approaches that do not incur contact information overhead, identify their shortcomings and state how this paper contributes in addressing the issue. Our first contribution is detailed in Section 3. First, we model contact information overhead and investigate its impact on data exchange success and energy consumption on portable handheld devices. Then we present the need for algorithms that reduce contact information overhead for OppNets in emerging IoT scenarios without suffering the shortcomings of existing solutions. In Section 4, we detail our second contribution, which addresses this. In particular, we adopt the combination of location-based and encounter-based forwarding approaches to propose the PoiFord algorithm. In Section 5, we validate our proposed model for contact information overhead and evaluate the performance of our proposed PoiFord through simulation experiments. Finally, Section 6 concludes this paper and discusses future work.

## 2. Problem Background

Most pioneering work on OppNets focused on the point-to-point (or destination-based) communication model, in which endpoints are identified by their ID. Basically, message headers include destination identifiers in order to ensure delivery through suitable relays. In recent years, however, the increasing use of the Internet for sharing information has directed research efforts towards disseminating and retrieving content, rather than connecting node pairs. In information (or content) centric communication [30], the content, rather than the nodes involved, are named. Interested parties (or subscribers) request content by their names, and the network is tasked with locating the sources (or publishers) and routing the content to the receivers. Contributions have also emerged in the aspect of location-based information sharing, e.g., floating content [31], where content is associated with a particular geographic area. However, this destination-less communication model is more suitable for content-centric applications where data is disseminated based on user interest (or feedback as in ODD [32]). In order to be a feasible backhaul solution for UANs, available resources need to be utilized sparingly while guaranteeing acceptable throughput. In other words, OppNets require means of determining suitable relays (among the multitude of nodes in Smart City scenarios) that can route messages to their respective gateways within a given time-to-live (TTL), thereby making the destination-based approach a more suitable communication model.

Different routing strategies following the destination-based communication model have been proposed over the years. There are routing strategies that do not require knowledge about the network (e.g., Epidemic [33]), thereby making them easy to implement. However, the tendency to consume excessive resources makes them unsuitable for sensed data collection. Utility-based protocols on

the other hand, require knowledge about the network to select suitable relay nodes during message forwarding. The ability for a node to deliver a message is usually determined by computing its “forwarding utility” for the destination, such that a higher value indicates a higher contribution towards message delivery (i.e., more chances of encountering better relay nodes or the destination itself).

Researchers adopt different strategies for computing forwarding utilities. There are strategies that predict future encounters between two nodes from the similarity of their movement patterns (e.g., MobySpace [34]). However, in sensed data collection, gateways nodes are static and do not exhibit similar movement patterns with suitable relay nodes. The delivery ability of nodes can also be determined from social characteristics (e.g., PeopleRank [35], Bubble Rap [36] and dLife [37]). Some protocols (such as CiPRO [38]) go a step further by utilizing contextual information—user information (e.g., email address, work and home address, occupation, mobility patterns, and communities they belong to) and device information (e.g., battery level and storage capacity)—to make forwarding decisions based on a profile match. However, the fact that gateway nodes neither portray social characteristics (e.g., belonging to social communities) nor possess enough contextual information limits the applicability of these strategies. Forwarding strategies can also be realized from encounter-based properties. Some strategies derive forwarding utilities from the number of encounters between nodes, so that messages are forwarded to nodes that encounter the destination (or its neighbors) more frequently (e.g., PROPHET [39], I-PROPHET [40] and ISW [41]). Others maintain a timer for each encountered node and determine the forwarding utility based on how recently the destination was encountered (e.g., Spray and Focus [42], TMS [43] and OPF [44]). By maintaining and updating encounter history about other nodes in the network, these approaches can efficiently identify routing paths to destination nodes [36], thereby making them more suitable for sensed data collection.

PROPHET in particular inspired most of the encounter-based protocols and continue to inspire forthcoming ones, and has been incorporated into the reference implementation maintained by the Internet Research Task Force Delay Tolerant Networks Research Group. PROPHET has also been trialed in real-world situations during the Sámi Network Connectivity project [45] and is being further developed for the European Union’s Seventh Framework Programme project (namely Networking for Communications Challenged Communities) [46]. Hence, irrespective of how the forwarding utility is computed, most encounter-based protocols apply PROPHET’s approach of exchanging and updating contact information. This section describes this approach and the problem of contact information overhead that arises. Existing solutions for addressing the problem and their shortcomings are also presented.

### 2.1. Overhead in Exchanging Contact Information

The PROPHET routing approach is as follows. When two nodes encounter each other, they exchange summary vectors that contain: (i) a message vector, which is a list of message identifiers in their buffers; and (ii) a contact vector, which is a list of every known node (i.e., directly encountered nodes as well as their neighbors) and the corresponding forwarding utility for each node. With the information received from the contact vector (i.e., the contact information), each node updates its knowledge about the network by re-computing forwarding utilities for nodes in its list as well as for new ones provided by the other node. It is after this process that the nodes in contact decide which messages to request from the other.

Certainly, an amount of overhead is incurred in exchanging contact information, which varies directly with the number of nodes and encounters in the network. First, since known nodes do not necessarily have to be directly encountered, the size of the list containing contact information eventually converges with the total number of nodes in the network [47]. Second, the overhead incurred in exchanging the information in this list (i.e., sending plus receiving) increases with the number of encounters in the network. Although portable handheld devices of recent times may possess enough storage space and processing capabilities (to maintain and update contact information, respectively), insight regarding the impact of contact information size on data exchange success and



energy consumption is lacking in literature. This leads to our first contribution, in which we model contact information overhead and then investigate its impact on data exchange success and the energy consumption on portable handheld devices.

## 2.2. Location-Based Solutions and Their Shortcomings

Location-based forwarding approaches do not require nodes to exchange knowledge upon encounter, hence, contact information overhead is not incurred. They exploit the concept of “home” [29], which is based on the observation that nodes tend to stay around a particular location for longer periods of time. Some approaches (e.g., HERO [29], A-HERO [48] and HERO++ [49]) partition the geographical area of the network into smaller regions and identify the home-region of nodes. Justified by the phenomenon of spatial locality [50–52], the chances of delivering a message are improved by forwarding it to nodes whose home-regions are increasingly closer to that of the destination. LOC [53] also considers the angular direction and distance of nodes to the approximate location of the destination.

Unfortunately, location-based forwarding approaches present drawbacks in terms of delivery guarantees. First, there is the issue of determining the best forwarder among nodes presenting identical characteristics, e.g., in HERO, nodes having the same home region as the destination, and in LOC, nodes located at similar distances to the destination. Simply put, the chances of locating the destination reduce with increasing node population. A-HERO addresses this by flooding the message among home nodes. In high node population, however, this leads to undesirable overheads and persistent buffer overflows that may eventually reduce throughput.

Second, nodes that have good encounter history or social relationship with the destination are not often identified through location-based forwarding utilities alone [54]. In fact, nodes that have good data transfer opportunities may not come from the same home-region—take office colleagues or classmates, for instance—and similarity in node movement patterns does not always guarantee an encounter between them—take users that have never met but living in neighboring streets, for instance. Therefore, while location-based approaches may be able to carry messages spatially closer to their respective destinations, additional mechanisms are required to further improve throughput. This brings about our second contribution, in which we propose a forwarding algorithm that can be incorporated into existing encounter-based routing protocols to reduce contact information overhead without reducing throughput.

## 3. A Model for Contact Information Overhead

In this section, we model contact information overhead and investigate its impact on data exchange success and energy consumption on portable handheld devices. We also present the need for algorithms that reduce contact information overhead for OppNets in emerging IoT scenarios.

### 3.1. Contact Information Size

While the amount of information exchanged between nodes in order to compute forwarding utilities may vary across protocols, the structure is the same as that of PROPHET. A node that runs on PROPHET maintains a set of contact information,  $C$ , for every encountered node,  $h$ . Let  $C = \{c_1, c_2, c_3, \dots, c_{n-1}\}$  be the set of contact information maintained at a node running on PROPHET. Each contact information  $c_h \in C$  is a tuple in the form of  $c_h = \langle h, u_h \rangle$ , where  $u_h$  is the node’s forwarding utility (which PROPHET’s authors refer to as “delivery predictability”) for an encountered node  $h$ . For analytical purposes,  $h$  and  $u_h$  are assumed to be a string and a double data type of  $X_S$  and  $X_D$  bytes, respectively. The size of  $c_h$ , a piece of contact information, becomes  $X_S + X_D$  bits. The transitive property of PROPHET allows a node to maintain delivery predictabilities for nodes it has never encountered, which are determined from contact information received from neighbors that have previously encountered them. This hastens the rate at which contact information is disseminated within the network. Consequently, the number of nodes for which contact information is maintained

eventually reaches  $n - 1$  (i.e.,  $|C| = n - 1$ ), where  $n$  is the total number of nodes in the network. The size of the contact information maintained at each node is given by Equation (1).

$$S_C = (n - 1)(X_S + X_D) \text{ bytes} \quad (1)$$

For example, consider the Universiti Teknologi Malaysia (UTM) campus which is located on 11.45 KM<sup>2</sup> of land [55] and registers about 18,000 students (as of 2017) [56]. It is fair to say that every student owns a portable handheld device. An IoT scenario would also consist of other nodes such as smart vehicles, sensors, access points and appliances equipped with short-range wireless communication interfaces. So let us assume that 80% of the students live in-campus and other nodes besides portable handheld devices account for 40% of the total node population in the scenario. Then, the number of portable handheld devices and other nodes becomes 14,400 and 6000, respectively, making a total of 24,000 networking nodes. Taking  $X_S$  and  $X_D$  as 10 bytes (i.e., a string of 5 characters) and 8 bytes, respectively, a portable handheld device would eventually maintain about 0.432 MB of contact information (using Equation (1)). Hence, portable handheld devices would have to send and receive a total of about twice this amount (i.e., 0.864 MB) of contact information whenever they encounter each other.

### 3.2. Impact of Contact Information Size on Message Forwarding

Here, we model the impact of contact information size on message forwarding in terms of delay (i.e., the amount of time remaining for forwarding messages after exchanging contact information) and energy consumption (i.e., the amount of energy required to send and receive this information per encounter).

#### 3.2.1. Delay Incurred in Exchanging Contact Information

Due to short-lived encounters in OppNets, it is important to forward enough data messages to the relay node before the link is disconnected. Since summary vectors need to be received before messages can be forwarded, the size of a summary vector needs to be as small as possible. Larger summary vectors would require more time to be transferred, and if the encounter duration is relatively short, some messages in the buffer may fail to be forwarded before the transmission opportunity is lost. The contact vector tends to account for a significant portion of the summary vector in terms of size, since it increases with node population and number of encounters. As a result, a reasonable amount of the delay incurred in exchanging summary vectors emanate from the size of contact vectors. Hence, less contact information contributes to higher data exchange success and vice versa. With a data transmission rate of  $R$  bytes per second on resource constrained nodes, the time,  $T_C$ , taken for a successful exchange of contact information is given by Equation (2).

$$T_C = \frac{2S_C}{R} \quad (2)$$

To illustrate this, consider two nodes, A and B, each with 625 KB of messages in their buffer. Supposing summary vectors do not need to be exchanged, a usable encounter duration of 5 s (i.e., the available time after removing delay in discovering encounter events and establishing connection) should be sufficient for a successful data transfer at the rate of 250 KB per second (in an ideal state). In other words, the nodes are able to exchange 1250 KB of data (i.e., each node is able to send 625 KB and receive 625 KB), hence, 100% data exchange success. If they were to exchange 10 KB of contact information before actual data transfer, the remaining encounter duration (i.e.,  $5 \text{ s} - T_C = 4.92 \text{ s}$ ) would permit a maximum of 1230 KB of data to be transferred, hence, 98.4% data exchange success. Table 1 shows the data exchange success for different amounts of contact information.

**Table 1.** Data exchange success for different amounts of contact information.

Contact Info. Size	Remaining Encounter Duration	Max. Transferable Data	Data Exchange Success
0 KB	5.00 s	1250 KB	100%
10 KB	4.92 s	1230 KB	98.4%
20 KB	4.84 s	1210 KB	96.8%
30 KB	4.76 s	1190 KB	95.2%
40 KB	4.68 s	1170 KB	93.6%
50 KB	4.60 s	1150 KB	92.0%
0.432 MB	0.00 s	0 KB	0.00%

### 3.2.2. Energy Utilized in Exchanging Contact Information

When two nodes encounter each other, they exchange summary vectors that contain  $C$ , the set of contact information. The energy expended in exchanging contact information per encounter,  $E_x$  Joules, is the sum of energy consumed in sending  $S_{tx}$  bits,  $E_{tx}$  Joules, and receiving  $S_{rx}$  bits,  $E_{rx}$  Joules. As shown in Equation (3), we assume that  $S_{tx} = S_{rx} = S_C$  bits, since the number of nodes for which contact information is maintained eventually converges with the total number of nodes in the network.

$$E_C = S_C(E_x) = S_C(E_{tx} + E_{rx}) \quad (3)$$

We assume that node batteries are not recharged until an observation period of  $T$  seconds has elapsed. The energy consumed within this period,  $E_C(T)$  Joules, is obtained by multiplying the energy expended in exchanging contact information for an encounter,  $E_C$  Joules, by the number of encounters within the period,  $N_T$  (cf. Equation (4)).

$$E_C(T) = E_C N_T \quad (4)$$

Take the energy consumed to send or receive a byte of data through the available wireless communication interface as  $E_{tx}$  Joules and  $E_{rx}$  Joules, respectively. Also, take the energy on a portable handheld device battery as  $E_B$  Joules, and the percentage users are willing to allocate to the OppNet as  $P_A\%$  (i.e., allocated energy =  $E_B \times P_A/100$  Joules). The percentage of allocated energy utilized in exchanging contact information after an observation period of  $T$  seconds is given by Equation (5).

$$P_E = \frac{E_C(T)}{P_A E_B} \times 10^4 \quad (5)$$

Again, consider the UTM campus IoT scenario with a total of 24,000 networking nodes. The number of encounters a node experiences depends on its popularity in the network. Let us assume that for a wireless transmission range of 10 m, the average number of encounters (with repetition) experienced by a node during an observation period of 24 h is 500. Since smartphones are energy constrained due to various applications running on them, we assume that users would be willing to allocate only an unnoticeable amount of their battery to the OppNet. Let us take this value as 3%. Let us also assume that devices communicate through Bluetooth (v4.1) and consume 1.225 micro Joules to send or receive a byte of data [57]. From Equation (5), 38.5% of the allocated energy on a smartphone battery of 12.705 Wh or 45,738 Joules (e.g., 3.85 V and 3300 mAh for SAMSUNG Galaxy J7) would be utilized in exchanging contact information alone. In that case, only 61.5% of the allocated energy would be remaining for other network operations, which include device discovery, exchanging other portions of the summary vector, sending and receiving data messages in the device buffer and computations! Table 2 shows the percentage of allocated energy utilized in exchanging contact information for a different number of encounters.



**Table 2.** The percentage of energy allocated to the OppNet that is utilized in exchanging contact information alone for a different number of encounters within 24 h (battery energy = 45,738 Joules; and allocated energy = 1372.14 Joules).

No. of Encounters	Energy Utilized (Joules)	% of Allocated Energy Utilized	% of Allocated Energy Remaining
300	317.2	23.1%	76.9%
400	423.0	30.8%	69.2%
500	528.7	38.5%	61.5%
600	634.5	46.2%	53.8%
700	740.2	53.9%	46.1%
800	846.0	61.7%	38.3%
1295	1369.5	99.8%	0.2%

### 3.3. Need for Algorithms that Reduce Contact Information Overhead

One of the challenges facing OppNet research for emerging IoT scenarios is the difficulty in obtaining rich datasets that portray human movement in urban scenarios. For instance, a campus scenario may include thousands of portable handheld devices on people (e.g., students, lecturers and buses) and endpoints (e.g., vehicles and access points) more densely packed and moving into transmission range more often than depicted in existing datasets. The impact of certain overheads may go unnoticed in a small and highly dense population—for instance, the overhead each node incurs due to flooding messages in a small population may not be significant when compared with a larger population. This is also the case for a large population with low density, since nodes may not encounter each other frequently enough. Hence, for the purpose of evaluating OppNet solutions in such scenarios, it may be more realistic to study human movement properties in real-world traces and reproduce them in synthetic movement models, until rich city-wide datasets are available. At the same time, researchers need to have this in mind while setting up synthetic movement scenarios.

The need to reduce contact information overhead is acknowledged only when the node population and dynamicity of the utilized movement scenario begins to resemble that of real-world urban environments. Contact information size raises two main issues with regard to routing performance. The first issue is regarding the portion of the encounter duration the process of exchanging summary vectors occupies. Considering the operations involved in neighbor discovery and link set up, contact information needs to be small enough to fully utilize the typically short encounter durations. The second issue is that of energy consumed in exchanging contact information. Since there is no guarantee that users will always be willing to shed all the available energy on their devices for the sake of the technology, the logical direction for OppNets is to minimize energy consumption.

Clearly, the size of contact information in summary vectors needs to be minimized as much as possible. The optimization provided by the Delay Tolerant Networking Research Group in the latest version of PROPHET's specification [58] suggests that contact information for nodes with delivery predictabilities less than a certain threshold be removed. However, the threshold requires careful selection for each node such that it has to be less than delivery predictability values usually present in the network for destinations for which the node is a forwarder. The authors also suggest that the threshold could be calculated based on delivery predictability ranges and the amount they change historically. Certainly, determining this threshold introduces additional complexities whose impact on performance the authors are yet to investigate. Until then, forwarding algorithms that can guarantee equal or acceptable throughput with less contact information are key, and are the subject of the next section.

## 4. An Algorithm for Reducing Contact Information Overhead

Message forwarding in high node population and dynamicity requires a means of minimizing contact information overhead without throughput degradations. We already learned from a previous study [47] that it may not be necessary to maintain the history of every encounter in order to realize

comparable throughput. The modification of PROPHET to maintain encounter history for only nodes from the same home-region traded only 11% throughput for about 84% average reduction in contact information maintained at each node. Hence, the challenge in optimizing the trade-off between contact information overhead and throughput lies in: determining which encounters to consider in terms of maintaining encounter history, and which ones to forego; and a forwarding approach that can utilize the available information to carry messages close enough to the destination or to nodes that have recorded their encounter history with the destination. Without a proper understanding of the relationship between node movement and encounter opportunities, important encounters may be neglected and the resulting insufficiency in knowledge may cause throughput degradation. In this section, we present PoiFord, and tackle the following points in the process:

- A means of minimizing routing information without significant throughput degradations, more specifically, addressing the challenge of determining the most relevant encounters for which to maintain encounter history; and
- A forwarding approach that can utilize the available information to carry messages close enough to the destination or to nodes that have encounter history with the destination.

By addressing these points, PoiFord is able to achieve the features shown in Table 3.

**Table 3.** Relevant forwarding solutions and their main features.

Forwarding Solution	Low Contact Info. Overhead	No Message Flooding	Can Determine Better Forwarders among Home Nodes	Does Not Infer Encounters from Location-Based Information Alone	Improves Delivery Guarantees
HERO [29]	✓	✓	×	×	×
A-HERO [48]	✓	×	×	×	×
HERO++ [49]	✓	✓	✓	×	×
LOC [53]	✓	✓	✓	×	×
Our proposal (PoiFord)	✓	✓	✓	✓	✓

#### 4.1. Overview of PoiFord

Our model in Section 3 showed that maintaining encounter history about more nodes leads to more contact information overhead, which is in turn directly proportional to the delay incurred in exchanging contact information and the energy utilized in the process. Therefore, less contact information implies less delay and energy consumption. The only problem is how to achieve this without degrading throughput. PoiFord is focused on reducing the amount of encounter history maintained by nodes so that less contact information would be exchanged during encounters. The basic idea behind the approach is as follows. Users often return to their points-of-interest (POIs) in which they may encounter other users that visit the same location regularly. Thus, there is a high chance that most well-connected nodes have at least one mutual POI (e.g., users living in the same house or working in the same office). PoiFord aims to minimize contact information by requiring only nodes that have mutual POIs to maintain encounter history for each other. By limiting information to be maintained only for neighbors with mutual POIs, the problem of existing location-based solutions (i.e., inferring future encounters from location-based information alone) can be addressed while the forwarding ability of nodes that are less connected to the destination can be inferred from location-based information, specifically through spatial locality. That way, nodes moving towards a destination that is located in a shopping mall would carry a message and forward it to a node that has a mutual POI with the destination (i.e., a node whose user works in a shopping mall) to deliver it. Hence, the task of reducing contact information overhead without degrading throughput can be divided into three subtasks:

1. Determining a node's significant locations or POIs.
2. Determining nodes for which to maintain encounter history based on mutual POIs.
3. Determining a forwarding utility that can deliver messages with the available information.

This section overviews PoiFord by presenting the tasks involved in realizing the three subtasks. As shown in Figure 1, the subtasks are addressed in three phases.

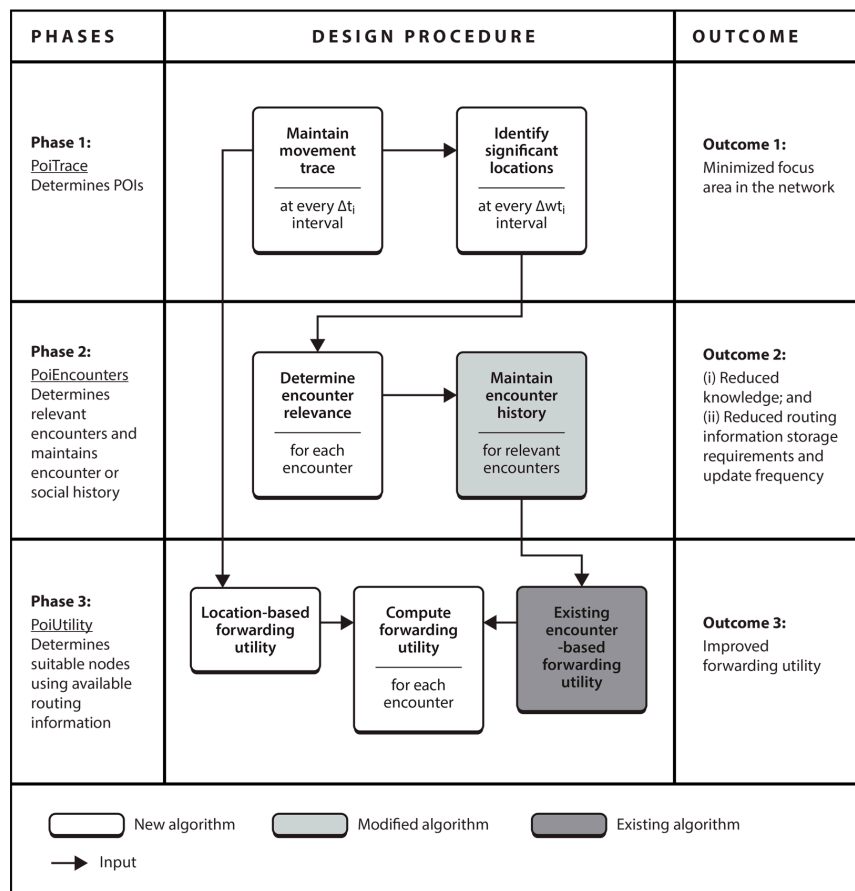


Figure 1. Functional block diagram of PoiFord.

**Phase 1.** The first phase is concerned with determining POIs. The current location is collected via GPS at fixed intervals and maintained in device memory. From this, significant locations are determined at larger fixed intervals. The major challenge in this phase is how to maintain GPS coordinates from which significant locations can be determined, knowing that the same pair of coordinates may never be recorded twice. To address this, we propose PoiTrace, a mechanism that maintains incoming GPS locations in a location table after converting them to a more stable form we call location references, from which POIs are determined. This minimizes the focus area for each node, as compared to the entire network area.

**Phase 2.** The second phase is concerned with identifying nodes for which contact information needs to be maintained in the form of encounter history. POI information is included in summary vectors, which nodes first exchange when they encounter each other. This allows nodes to determine relevant encounters with respect to maintaining contact information. Concerned nodes keep record of an encounter only when they have mutual POIs. The major challenge in this phase is how to identify mutual POIs of a node pair, knowing that the likelihood of finding exactly the same set of GPS locations on two nodes is low. To address this, we propose PoiEncounters, a mechanism that seeks for mutual POIs of encountered node pairs, and if they are found, maintains a history of the encounter without any transitive property. This minimizes the amount of contact information and knowledge required to make forwarding decisions. In addition, since nodes are only concerned with relevant encounters, the frequency of updating contact information is reduced. PoiEncounters is flexible in the sense that it can be incorporated into existing encounter-based routing protocols.

**Phase 3.** The third phase is concerned with determining suitable nodes to forward the message. Without a transitive property, and since nodes maintain encounter history of only relevant encounters, knowledge about the network is limited. Hence, the major challenge in this phase is determining suitable nodes with less knowledge about the network. To address this, we propose PoiUtility, a forwarding utility that determines nodes that either have better chances of delivery or can make better progress towards the destination. The forwarding utility is acquired from two utilities: one that determines suitable relays from the available encounter history; and another that determines nodes capable of carrying the message spatially closer to the destination, in order to compensate for the lack of a transitive property.

The following assumptions are made in the design of PoiFord: (i) each node is a smart mobile device and is equipped with a Global Positioning System (GPS); (ii) nodes are collaborative and willing to participate in routing; and (iii) source nodes have the necessary information for destinations, which are node ID and location-based information in this case.

#### 4.2. PoiFord Design

We present the design of PoiFord by detailing each of the phases presented in Section 4.1. This section is organized as follows. Section 4.2.1 presents PoiTrace, a mechanism for locally identifying significant locations (i.e., POIs) from GPS information obtained on the go. With the aid of these POIs, Section 4.2.2 proposes PoiEncounters, a mechanism for maintaining encounter history that does not increase or become stale in time. Section 4.2.3 proposes PoiUtility, a forwarding utility based on the obtained location-based information and available encounter history. The forwarding algorithm for PoiFord using the proposed forwarding utility is presented in Section 4.2.4.

##### 4.2.1. Phase 1: Identification of Significant Locations

Phase 1 presents the design of PoiTrace, for identifying POIs. User movement may reveal multiple significant locations and can be exploited for routing. However, we are interested in investigating how our POI approach can reduce contact information overhead. In order to keep the idea comprehensive, the mechanism proposed here identifies only the two most significant locations, which we refer to as “home” and “work” location. Apart from these locations often corresponding to the actual home and work locations of users, research also shows that most users have at least two most significant locations and regularly commute between them [59].

**Collecting and recording location information.** At every sampling interval  $\Delta t$ , each node collects its current position by GPS in form of latitude and longitude and records it in the corresponding time slot in the location table (cf. Definition 1)—note that PoiFord is fully distributed and does not require synchronization between devices. For example,  $(x_i, y_i)$ , the GPS coordinates collected at  $\Delta t_i$ , the current sampling interval, are recorded in the current time slot as  $loc_i$ , a location reference. Although GPS is globally available, the signal may not always be reliable in geographically restricted areas (e.g., in buildings and underground locations). In case the signal is too weak or lost, the current position is approximated as the previous record in the location table.

**Definition 1 (Location Table).** The location table,  $LT$ , which consists of  $n$  time slots, is a set of  $n$  elements, each known as a location reference (cf. Equation (6)). Each location reference  $loc_i \in LT$  is a tuple of the format  $loc_i = \langle x_i, y_i \rangle$ , where  $i \in \{1 : Z | i \in 1 \dots n\}$  indicates the current time slot.

$$LT = \{loc_1, loc_2, loc_3, \dots, loc_n\} = \{\langle x_1, y_1 \rangle, \langle x_2, y_2 \rangle, \langle x_3, y_3 \rangle, \dots, \langle x_n, y_n \rangle\} \quad (6)$$

The basic idea behind the algorithm for recording location information is as follows: location references in  $LT$  should be able to map geographical locations visited for longer periods from user movement, by representing them with circular areas. In order to achieve this, a certain extent of deviation between successive incoming GPS locations is tolerated while acquiring location references.

The location reference  $loc_a$  for any two locations  $(x_1, y_1)$  and  $(x_2, y_2)$  is the same if the circular areas formed by radius  $r$  from both locations intersect. This condition is fulfilled if the Euclidean distance between the two locations  $d_{1,2}$  is less than  $2r$  (cf. Equation (7)).

$$d_{1,2} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \quad (7)$$

Hence, if the circular area formed by  $r$  from any incoming pair of coordinates  $(x_i, y_i)$  intersects with the circular area formed by  $r$  from a previous pair of coordinates  $(x_a, y_a)$ , the corresponding location reference  $\langle x_a, y_a \rangle$  is formed from the existing coordinates. Otherwise, a location reference  $\langle x_i, y_i \rangle$  is formed from the incoming coordinates (cf. Equation (8) and Figure 2).

$$loc_i = loc(d_{a,i}) = \begin{cases} \langle x_a, y_a \rangle, & d_{a,i} < 2r \\ \langle x_i, y_i \rangle, & d_{a,i} \geq 2r \end{cases} \quad (8)$$

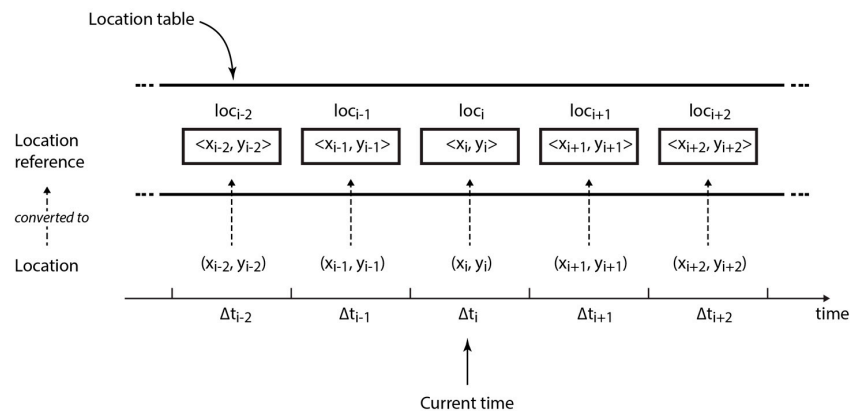


Figure 2. Structure of the location table.

Algorithm 1 summarizes how  $loc_i$  is acquired using the current location in the form of latitude  $x_i$  and longitude  $y_i$ , the previous location references in  $LT$ , and a threshold distance  $r$ . Consequently, the number of recurrences of a location reference  $loc_a$  in  $LT$  represents the number of periods in which the user is present in the circular area formed by  $r$  from location  $(x_a, y_a)$ .

---

**Algorithm 1** The algorithm for recording location information

---

**Input:**  $(x_i, y_i)$ , location references in  $LT\{loc_a\}$ ,  $r$

**Output:**  $loc_i$

```

1  foreach incoming location  $(x_i, y_i)$  in sampling interval  $\Delta t_i$  do
2      foreach Input  $\{loc_a\}$  do
3           $d_{a,i} = \sqrt{(x_a - x_i)^2 + (y_a - y_i)^2}$ ;
4          if  $d_{a,i} < 2r$  then
5               $loc_i = \langle x_a, y_a \rangle$ ;
6              break;
7          end
8          else
9               $loc_i = \langle x_i, y_i \rangle$ ;
10         end
11     end
12 end
13 return  $loc_i$ ;

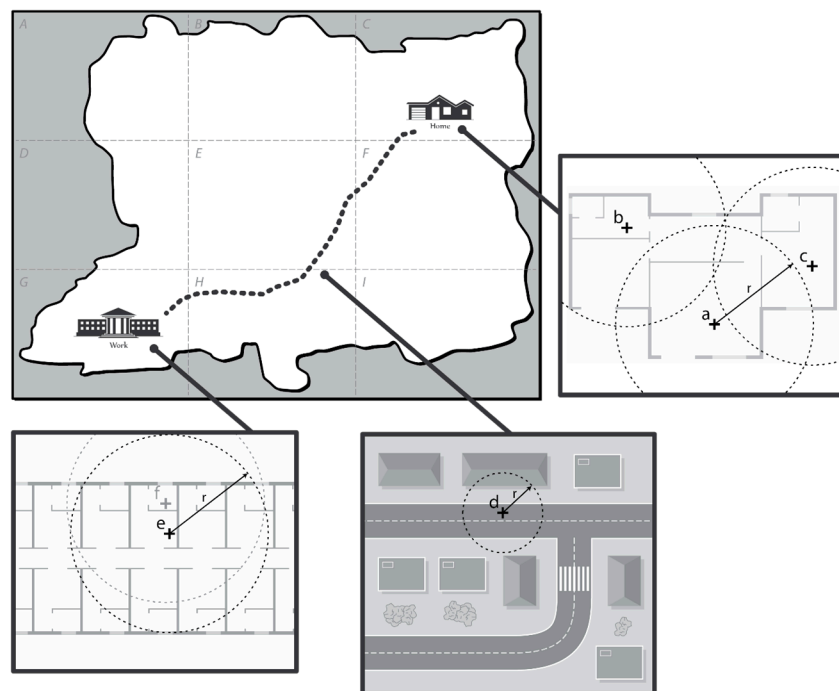
```

---



**Extracting significant locations.** Significant locations are identified by more number of recurrences in *LT*. At every sampling interval  $\Delta wt$ , a node running on PoiFord extracts the two most recurring location references from its location table—where  $w$  is a constant of the algorithm. These locations represent POIs, and are referred to as the node’s home and work locations,  $(x_h, y_h)$  and  $(x_w, y_w)$ , respectively. Although incoming GPS coordinates may slightly vary each time the node is in either location, the location reference records only a single pair of approximated coordinates each sampling interval  $\Delta t$ , provided the circular area formed by the incoming pair of coordinates intersects with that of a previous coordinate.

Here, to facilitate understanding, we further explain the working principle of PoiTrace with an example scenario. Consider the map of a fictional city in Figure 3 which is divided into 9 regions labelled *A* to *I*. Charlie, a user in the city, lives in region *C* and travels to work in region *G*. The first three GPS readings take place in his house at location *a*, *b*, and *c*. The next reading takes place on his way to work, at location *d*. The next two readings take place in his office, at location *e* and *f*.  $(x_a, y_a)$ , the GPS reading in location *a* is recorded as  $\langle x_a, y_a \rangle$  in the location table. However, as shown in Table 4, the readings at location *b* (i.e.,  $(x_b, y_b)$ ) and *c* (i.e.,  $(x_c, y_c)$ ) are also recorded as  $\langle x_a, y_a \rangle$ , instead of  $\langle x_b, y_b \rangle$  and  $\langle x_c, y_c \rangle$ , respectively. This is because the circle of radius  $r$  formed from these locations intersects with the circle formed from a previous location, i.e., location *a*. Likewise, the readings at *e* and *f* are both recorded as  $\langle x_e, y_e \rangle$ . After a period of  $\Delta 6t_1$  (here we take  $w$  as 6), the two most significant locations of this user become  $(x_a, y_a)$  and  $(x_e, y_e)$ .



**Figure 3.** Map of a fictional city.

**Table 4.** Recording location information in the location table with PoiTrace.

Time Slot	Location	GPS Coordinates	Location Reference
$\Delta t_1$	<i>a</i>	$(x_a, y_a)$	$\langle x_a, y_a \rangle$
$\Delta t_2$	<i>b</i>	$(x_b, y_b)$	$\langle x_a, y_a \rangle$
$\Delta t_3$	<i>c</i>	$(x_c, y_c)$	$\langle x_a, y_a \rangle$
$\Delta t_4$	<i>d</i>	$(x_d, y_d)$	$\langle x_d, y_d \rangle$
$\Delta t_5$	<i>e</i>	$(x_e, y_e)$	$\langle x_e, y_e \rangle$
$\Delta t_6$	<i>f</i>	$(x_f, y_f)$	$\langle x_e, y_e \rangle$

#### 4.2.2. Phase 2: Encounter Relevance and Maintaining Encounter History

In this section, we present PoiEncounters, the mechanism for determining relevant encounters and maintaining encounter history using the significant locations identified in the previous section. Without any loss of generality, and to demonstrate how PoiFord can be incorporated into an existing routing protocol, PoiEncounters adopts PROPHET's approach for acquiring and computing encounter history, with two major modifications (Note that besides PROPHET's, algorithms of other existing encounter-based routing protocols can be utilized. This demonstrates the flexibility of PoiFord). Unlike PROPHET, which maintains encounter history for every encountered node, encounter history between  $a$  and  $b$ , a node pair, is maintained if and only if at least one of the significant locations of  $a$ 's intersects with at least one of  $b$ 's. Also, PoiEncounters does not utilize the transitive property.

**Determining encounter relevance.** Upon encounter, two nodes  $a$  and  $b$  exchange summary vectors containing their significant locations, which in this case are their home and work locations, as well as the encounter history stored at each node. With this information, each node determines if the circular area of radius  $r$  formed from either of its significant locations intersect with the circular area formed from either of the other's. Given their home location as  $home_a$ ,  $home_b$  and their work location  $work_a$ ,  $work_b$ , respectively, each node determines if  $home_a$  intersects with  $home_b$  or  $work_b$ , and if  $work_a$  intersects with  $home_b$  or  $work_b$ . Let  $\{x_{sA}, y_{sA}\}$  represent the coordinates of  $a$ 's home or work location, and  $\{x_{sB}, y_{sB}\}$  represent the coordinates of  $b$ 's home or work location. If there is an intersection (i.e., if  $d_{sA,sB} < 2r$ ),  $a$  maintains/updates its encounter history for  $b$ , and  $b$  does the same for  $a$ . The idea is to find nodes that share the same significant location.  $2r$  (i.e., the diameter of the circle) represents the span of the area of concern, which may either be a living home or an office.

**Maintaining and updating encounter history.** The encounter history table holds encounter records in the form of node ID and delivery predictability [39]. Updating the encounter history table is done in two steps. First, each node checks its encounter history table if an encounter history for the other node  $e_{(a,b)}$  already exists. Then the necessary PROPHET operations are carried out as follows: if a history exists, it is aged based on  $k$ , the number of time units elapsed since the last encounter, as shown in Equation (9), where  $\gamma \in [0, 1]$  is the aging constant; otherwise, it is assigned a value of 0.

$$e_{(a,b)aged} = e_{(a,b)} \times \gamma^k \quad (9)$$

With this value, the encounter history (or new encounter history if there was a previous encounter) is computed using Equation (10), where  $e_{init} \in [0, 1]$  is an initialization constant.

$$e_{(a,b)} = e_{(a,b)aged} + (1 - e_{(a,b)aged}) \times e_{init} \quad (10)$$

After computing the encounter history for the node pair, the encounter history table is updated by inserting this value (or by replacing the old value if there was a previous encounter). Algorithm 2 summarizes the process of updating the encounter history table when two nodes encounter each other.

**Algorithm 2** The algorithm for updating the encounter history table with PoiEncounters

---

**Input:** encounter history table,  $a$ 's significant locations  $\{x_{sA}, y_{sA}\}$ ,  $b$ 's significant locations  $\{x_{sB}, y_{sB}\}$ ,  $r$ ,  $\gamma$ ,  $k$ ,  $e_{init}$   
**Output:** updated encounter history table

```

1  foreach Input  $\{x_{sA}, y_{sA}\}$  do
2      foreach Input  $\{x_{sB}, y_{sB}\}$  do
3           $d_{sA,sB} = \sqrt{(x_{sA} - x_{sB})^2 + (y_{sA} - y_{sB})^2}$ ;
4          if  $d_{sA,sB} < 2r$  then
5              if  $e_{(a,b)}$  can be found in encounter history table then
6                   $e_{(a,b)aged} = e_{(a,b)} \times \gamma^k$ ;
7              end
8              else
9                   $e_{(a,b)aged} = 0$ ;
10             end
11              $e_{(a,b)} = e_{(a,b)aged} + (1 - e_{(a,b)aged}) \times e_{init}$ ;
12             Insert  $e_{(a,b)}$  into encounter history table;
13             break;
14         end
15     end
16 end
17 return updated encounter history table;
```

---

#### 4.2.3. Phase 3: Forwarding Utility

In this section, PoiUtility, the forwarding utility for PoiFord is determined using the available encounter history. Since nodes maintain encounter history for only a specific set of nodes (i.e., ones with which they share mutual POIs), neighboring nodes may not have any encounter history with the destination, especially when the destination's POIs are located in far-away districts. Hence, instead of a transitive property (which tends to increase contact information size with node population as nodes may eventually have to compute and maintain a forwarding utility for every other node in the network), geographical closeness to the destination is used to determine the forwarding ability of a node that has no encounter history with the destination. Encounter history with the destination is only considered when nodes that share a mutual POI with the destination are encountered. Therefore, PoiUtility, the overall forwarding utility, comprises of a location-based utility and an encounter-based utility.

**Measuring closeness-to-destination.** The information maintained in the encounter history table does not include a transitive property, i.e., a means of computing the likelihood of  $b$  to deliver a message to the destination  $d$  through  $c$ , based on the encounter history of  $b$  and  $c$ , and that of  $c$  and  $d$ . Hence, the location-based "closeness-to-destination" utility is used to select a node that can take the message closer to the destination, and possibly improve the likelihood of forwarding it to nodes that have an encounter history with the destination. The main conditions are that for each message: (i) the location of the destination node,  $d$ , denoted by  $(x_d, y_d)$ , is known by the source node—e.g., each sensor node can be preconfigured with this information during network initialization (PoiFord is designed for collecting sensed data from sensors—which may either be mobile (e.g., mobile user devices) or static (e.g., sensors deployed along roadsides) to their respective static gateways (e.g., an access point in a shopping mall) in Smart City scenarios. Each sensor node can be preconfigured with the location of the destination node(s) during network initialization. In case a destination node needs to be moved or replaced, the new information can be disseminated to the sources through the network); and (ii) the source node can compute the Euclidean distance from its location,  $(x_s, y_s)$ , to the

destination node's location, denoted by  $d_{s,d}$ . With this information available in the message header, the closeness of a node  $a$  to the destination,  $d$ , denoted by  $dMin_{a,d}$ , can be determined. The procedure starts by computing  $(\bar{x}_a, \bar{y}_a)$  as shown in Algorithm 3, which is the centroid of the location references in  $a$ 's location table,  $LT_a$ . In other words,  $\bar{x}_a$  and  $\bar{y}_a$  are the mean values of the  $x$  and  $y$  coordinates of every location reference in  $LT_a$ , respectively. Then  $dMin_{a,d}$  is given by the Euclidean distance between  $(\bar{x}_a, \bar{y}_a)$  and  $(x_d, y_d)$ . Note that for a static source node,  $s$ ,  $dMin_{s,d} = d_{s,d}$ , i.e., the closeness-to-destination utility of  $s$  is equivalent to the Euclidean distance from  $s$  to the destination node,  $d$ .

---

**Algorithm 3** The algorithm for acquiring  $(\bar{x}_a, \bar{y}_a)$

---

**Input:** location references in  $LT\{loc_a\}$   
**Output:**  $(\bar{x}_a, \bar{y}_a)$

```

1   $i = 0;$ 
2   $\bar{x}_a = 0;$ 
3   $\bar{y}_a = 0;$ 
4  foreach Input  $\{loc_a\}$  do
5       $\bar{x}_a = \bar{x}_a + x_a;$ 
6       $\bar{y}_a = \bar{y}_a + y_a;$ 
7       $i = i + 1;$ 
8  end
9   $\bar{x}_a = \bar{x}_a / i;$ 
10  $\bar{y}_a = \bar{y}_a / i;$ 
12 return  $(\bar{x}_a, \bar{y}_a);$ 

```

---

**Overall forwarding utility.** The overall forwarding utility (i.e., the PoiUtility) of a node is given by summing the weight for encounter history and the weight for closeness-to-destination. The former can be computed for nodes that have an encounter history with the destination node, while the latter can be computed for every node in the network. That way, nodes that do not have an encounter history with the destination node may still have some degree of forwarding utility through the closeness-to-destination weight. This allows them to carry the message spatially closer to the destination node, thereby increasing the chances of encountering and forwarding it to nodes that have a higher likelihood of delivery via the encounter history weight. In order to achieve these weights, encounter history and closeness-to-destination are multiplied by constants  $\mu$  and  $\omega$ , respectively, where  $\mu \gg \omega$ . The sum of both constants equals 1, thereby acting as a slider that decides how much impact each weight has on the overall forwarding utility (cf. Equation (11)).  $\mu$  is much greater than  $\omega$  in order to give more impact to the encounter history weight, so that messages are always directed towards nodes that have encounter history with the destination. To implement this, we select the value of  $\mu$  and  $\omega$  as 0.8 and 0.2, respectively.

$$\mu + \omega = 1 \quad (11)$$

The weight for encounter history, denoted by  $E_{(a,d)} \in [0, \mu]$ , at every node  $a$  for each known destination node,  $d$ , is given by Equation (12), while the weight for closeness-to-destination, denoted by  $D_{(a,s,d)} \in [-\infty, \omega]$ , is given by Equation (13). From Equation (13),  $D_{(a,s,d)}$  increases as  $dMin_{a,d}$  reduces, so that a node that often visits locations closer to the location of the destination node presents a higher closeness-to-destination weight. The overall forwarding utility, denoted by  $PoiUtility_{(a,s,d)} \in [-\infty, 1]$ , is given by Equation (14). Note that  $E_{(a,d)}$  is equal to 0 if node  $a$  does not have any encounter history with node  $d$ , in which case  $PoiUtility_{(a,s,d)}$  becomes  $D_{(a,s,d)}$ .

$$E_{(a,d)} = e_{(a,d)} \times \mu \quad (12)$$

$$D_{(a,s,d)} = \left(1 - \frac{dMin_{a,d}}{d_{s,d}}\right) \times \omega \quad (13)$$

$$PoiUtility_{(a,s,d)} = E_{(a,d)} + D_{(a,s,d)} \quad (14)$$

#### 4.2.4. Forwarding Algorithm

PoiFord forwards messages to suitable relays using the PoiUtility. The operation of PoiFord is as follows. When any node  $a$  with a list of messages,  $\{m_a\}$ , encounters another node,  $b$ , node  $a$  receives a summary vector from node  $b$  containing:

- List of message identifiers in node  $b$ 's buffer,  $\{m_b\}$ ;
- Node  $b$ 's POIs, i.e., home and work location,  $(x_h, y_h)$  and  $(x_w, y_w)$ , respectively (with which node  $a$  determines whether or not to maintain encounter history with node  $b$ —node  $a$  updates the previous value if one already exists);
- Node  $b$ 's encounter history table,  $\{ET_b\}$  (with which node  $a$  computes node  $b$ 's encounter history weight,  $E_{(b,d)}$ , for the destination node,  $d$ , of each message in  $\{m_a\}$ —this value is 0 for a destination node that node  $b$  has no encounter history with); and
- $(\bar{x}_b, \bar{y}_b)$  (with which node  $a$  computes the closeness-to-destination utility of node  $b$  for each message in  $\{m_a\}$ ).

With this information and others available in message headers (i.e., the destination node location,  $(x_d, y_d)$ , and the Euclidean distance from this location to the source node's location,  $d_{s,d}$ ), node  $a$  computes  $PoiUtility_{(b,s,d)}$  for each message in  $\{m_a\}$ . A straightforward replication technique is adopted onwards: node  $a$  compares  $PoiUtility_{(a,s,d)}$  and  $PoiUtility_{(b,s,d)}$ , then a copy of any message in its buffer for which it has a less PoiUtility is forwarded to node  $b$ , provided the message is not already in  $\{m_b\}$ . The strategy for selecting suitable relay nodes among a set of peers (i.e., nodes within transmission range) to forward messages is summarized in Algorithm 4.

---

**Algorithm 4.** Selecting suitable relay nodes for messages with PoiFord

---

**Input:** list of messages in buffer  $\{m_a\}$ , set containing list of messages in peers' buffers  $\{m_b\}$ , set of encounter history table of peers  $\{ET_b\}$ ,  $(\bar{x}_a, \bar{y}_a)$ , set of centroid of location references in peers' location tables  $(\bar{x}_b, \bar{y}_b)$

**Output:** list of suitable relay node against respective message  $\{B, M\}$

```

1  foreach Input  $\{m_a\}$  do
2      Obtain  $(x_d, y_d)$  from the header of  $m_a$ ;
3      Obtain  $d_{s,d}$  from the header of  $m_a$ ;
4       $dMin_{a,d} = \sqrt{(\bar{x}_a - x_d)^2 + (\bar{y}_a - y_d)^2}$ ;
5       $E_{(a,d)} = e_{(a,d)} \times \mu$ ;
6       $D_{(a,s,d)} = (1 - (dMin_{a,d}/d_{s,d})) \times \omega$ ;
7       $PoiUtility_{(a,s,d)} = E_{(a,d)} + D_{(a,s,d)}$ ;
8      foreach Input  $\{ET_b, (\bar{x}_b, \bar{y}_b)\}$  do
9           $dMin_{b,d} = \sqrt{(\bar{x}_b - x_d)^2 + (\bar{y}_b - y_d)^2}$ ;
10          $E_{(b,d)} = e_{(b,d)} \times \mu$ ;
11          $D_{(b,s,d)} = (1 - (dMin_{b,d}/d_{s,d})) \times \omega$ ;
12          $PoiUtility_{(b,s,d)} = E_{(b,d)} + D_{(b,s,d)}$ ;
13         if  $PoiUtility_{(b,s,d)} > PoiUtility_{(a,s,d)}$  then
14             if  $\{m_b\}$  does not contain  $m_a$  do
15                 Insert  $\{b, m_a\}$  into  $\{B, M\}$ ;
16             end
17         end
18     end
19 end
20 return  $\{B, M\}$ ;

```

---

## 5. Evaluation

This section presents the evaluation methodology and experimental results for the proposed model for contact information overhead and PoiFord.



### 5.1. Evaluation Methodology

In order to improve overall network performance, encounter-based forwarding approaches rely on contact information collected on the go to make forwarding decisions. Such information is often exchanged in summary vectors during encounters in order to remain up-to-date. We experiment with PROPHET routing protocol in order to show the impact of the overhead incurred during this exchange.

The performance of our proposed PoiFord is evaluated based on selected metrics by incorporating it into PROPHET and comparing the results before and after the incorporation. We also experiment with HERO to show that location-based forwarding approaches do not incur contact information overhead but may suffer low delivery guarantees. Configuration settings for PROPHET are in accordance with the values given by Lindgren et al. [39], which are 0.98 for  $\gamma$  and 0.75 for  $e_{init}$ . For PoiFord, the radius,  $r$ , for maintaining encounter-based routing information is taken as 10 m, and location-based information is maintained at hourly intervals in a location table of 24 time slots. HERO collects a GPS reading every hour for maintaining a location table of 7 days, and the region that records the most locations is considered the home region.

Since we evaluate routing performance through simulations, node mobility needs to portray realistic human movement properties. While real-world traces represent actual movement scenarios, they are less suitable for this evaluation due to: (i) the lack of realistic node density and encounter frequencies in city-wide scenarios; and (ii) the inability to change network properties such as node population and geographical area without interfering with the encounter opportunities between nodes. Hence, we resort to using a synthetic mobility model that provides the desired level of flexibility and is also able to reproduce realistic properties of human movement.

The remainder of this section is organized as follows. Section 5.1.1 presents the simulation set-up. In Section 5.1.2, metrics for evaluating the performance of PROPHET, PoiFord and HERO are presented. Then Section 5.1.3 models PoiFord's contact information overhead for theoretical analysis and comparison with PROPHET.

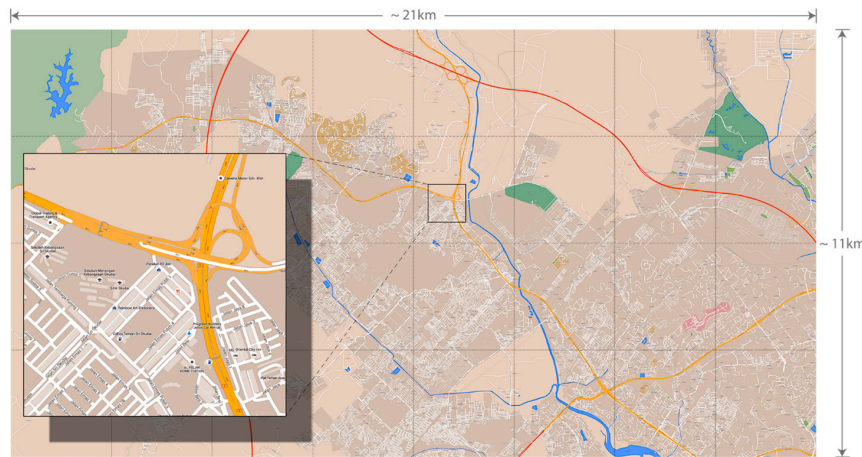
#### 5.1.1. Simulation Set-Up

The ONE simulator [60] is used to carry out simulations. We assume mobile users in a city using smartphones or similar handheld devices equipped with GPS and a Bluetooth interface operating at 2 Mbps with a transmission range of 10 m. The free buffer space of the nodes for routing-related tasks is limited to 10 MB, as users may not be willing to shed all their storage space to carry data on behalf of the technology. In order to observe a realistic node mobility and portray variations of human movement patterns that correspond to users of different occupations and backgrounds, the Working Day Movement (WDM) model [61] is used.

As shown in Figure 4, the simulation area roughly covers a 21 KM  $\times$  11 KM terrain, and consists of 32 districts. Since WDM models daily movement routines in working days, a total simulation duration of 5 days is chosen, which is distributed as follows: 1 day for warm-up (to ensure complete encounter and location history generation); 3 days for message generation and routing; and 1 day for cool-down. To simulate an OppNet for collecting data from sensors to their respective gateways in a city, static nodes are placed in popular locations to represent 64 sources and 32 destinations, respectively. Specifically, the source nodes were located in node homes (to represent residential areas) and along roads (for environmental sensors), while the destination nodes were located in other popular locations (to represent offices and shopping malls). Each source generates 1 message to a randomly selected destination every hour (to avoid bias), making a total of 4608 messages at the end of 3 days. Message TTL is set to 24 h, since focus is on messages to be delivered during a period of 1 day. The size of messages is uniformly distributed between 10 KB and 15 KB, considering the type of delay-tolerant applications mentioned in Section 1.1.

Energy consumption for Bluetooth is according to the configuration settings in the module proposed by Silva et al. [62]. All nodes are assigned an initial battery capacity of 4800 Joules, while the energy expended in receiving or sending messages is set to 0.08 mW/s. We are interested in only

energy consumed due to forwarding decisions made by each forwarding algorithm under comparison. Therefore, only energy consumed from receiving and sending messages is considered and other means through which nodes consume energy is ignored. On that basis, scan energy (i.e., energy consumed from device discovery), scan response energy (i.e., energy consumed from device discovery response), and base energy (i.e., energy consumed in idle state) is set to 0. The initial energy is high enough so nodes do not run out of energy during the simulation (Note that energy consumption in the simulation is mainly for performance comparison, and is not intended to represent realistic energy values). The parameters used for the simulation setup are shown in Table 5. 10 trials with different random seeds are simulated for each result in order to present the average and the 95% confidence interval.



**Figure 4.** The Skudai simulation area (map data provided by OpenStreetMap, 2015).

**Table 5.** Simulation parameters.

Parameter	Value
Total simulation time (days)	5
Warm-up period (days)	1
Cool-down period (days)	1
Wireless communication interface	Bluetooth
Transmission range (m)	10
Transmission rate (Mbps)	2
Buffer size (MB)	10
Message size (KB)	10 to 15
Message TTL (days)	1
Number of nodes	1349
Ave. message generation rate/node	1 message/hour
Battery capacity (Joules)	4800
Receive/transmit energy (mW/s)	0.08

### 5.1.2. Performance Evaluation Metrics

This section introduces the metrics for the performance evaluation, namely throughput, average delivery delay, message transmission overhead, and average energy consumption.

**Throughput.** As shown in Equation (15), throughput is the ratio between  $d$ , the total number of messages successfully delivered at their respective destinations, and  $g$ , the total number of messages generated. This signifies the message delivery efficiency of a forwarding algorithm, within the assigned TTL.

$$\text{Throughput} = d/g \quad (15)$$

**Average delivery delay.** Delivery delay is the time elapsed between message generation and delivery. As shown in Equation (16), the average delivery delay is the mean delivery delay in the network, where  $G_i$  and  $D_i$  are the generation time and the delivery time of the  $i$ th message, respectively. This gives an insight on how long it takes to deliver a message.

$$\text{Average delivery delay} = \sum_{i=1}^d (D_i - G_i) / d \quad (16)$$

**Message transmission overhead.** As shown in Equation (17), message transmission overhead represents the average number of transmissions required to deliver a message copy; where  $T$  is the total number of message transmissions, and  $V$  is the number of times messages were delivered. This is equivalent to the cost of delivering messages with a forwarding algorithm, since every transmission consumes energy on user devices.

$$\text{Message transmission overhead} = (T - V) / V \quad (17)$$

**Average energy consumption.** Energy consumption (in Joules) is the amount of initial energy expended at the end of the simulation. As shown in Equation (18), the average energy consumption is the mean energy consumed by nodes in the network, where  $n$  is the total number of nodes, and  $E_i$  and  $e_i$  are the initial energy and the remaining energy of the  $i$ th node, respectively. This gives an insight into how routing impacts resource consumption.

$$\text{Average energy consumption} = \sum_{i=1}^n (E_i - e_i) / n \quad (18)$$

### 5.1.3. Modelling PoiFord's Contact Information Overhead

The contact information overhead incurred by PoiFord is based on the size of the coordinates for POI and centroid information as well as the amount of encounter-based information maintained on a node. A pair of GPS coordinates occupies  $2X_D$  bits (i.e.,  $X_D$  for longitude and  $X_D$  for latitude). POI information, which is made up of home and office location, requires  $4X_D$  bits, while centroid information requires  $2X_D$  bits. This gives a total of  $6X_D$  bits.

For encounter-based routing information, the circular area represents the geographical span of the user's home or office. Hence, the radius of the circle,  $r$ , is chosen to reflect this. The choice of  $r$  therefore determines the amount of encounter history maintained by a node. A larger value includes more neighboring POIs and users such as a neighboring house or office, and vice versa. In real-world implementation, a user may be allowed to select this value or a mechanism could be introduced to automatically select this value based on the amount of resources the user is willing to allocate to the technology.

For analytical purposes, we take the average number of nodes for which encounter history needs to be maintained as  $(n-1)/v$  for home location and  $(n-1)/v$  for office location, where  $n$  and  $v$  represent the total number of nodes in the network and the number of POIs in the network, respectively. This gives a total number of  $2(n-1)/v$  nodes for which encounter history needs to be maintained. With  $X_S$  bits and  $X_D$  bits for holding node ID and the corresponding forwarding utility, respectively, a storage space of  $2(n-1)(X_S + X_D)/v$  is required for maintaining contact information. The size of contact information,  $S$ , maintained by PoiFord is given by Equation (18).

$$S = 2 \left[ 3X_D + \frac{(n-1)(X_S + X_D)}{v} \right] \text{ bits} \quad (19)$$

For a worst-case scenario of 100,000 nodes, 500 POIs, 80 bit string (i.e., 5 characters) for node ID and a 64 bit double for longitude, latitude, and delivery predictability, PoiFord generates a contact

information of only about 7.2 KB on each node, as compared with 1800 KB for PROPHET. This value remains stable if the ratio of node population to POIs remains roughly the same. Otherwise if the ratio increases, contact information size only increases by a small fraction. Contact information size can be reduced either by reducing  $u$  or the size of  $r$ . This reduces the number of GPS locations to be maintained in the location table, and the number of encounters for which to maintain history, respectively.

## 5.2. Results and Discussion

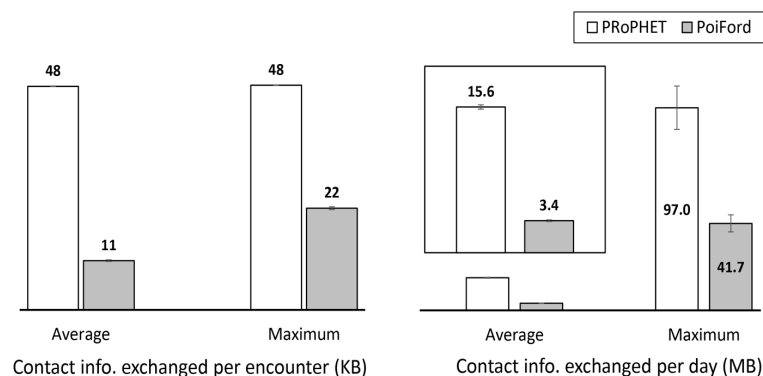
This section presents and discusses the performance evaluation results in two parts detailed in Sections 5.2.1 and 5.2.2. The first part (i.e., Section 5.2.1) presents results regarding contact information overhead. Specifically, simulation results about the amount of contact information maintained by PROPHET and the reduction achieved by the proposed PoiFord are presented. Then the impact of increasing network size on the amount of contact information maintained by PROPHET and PoiFord is presented and compared with theoretical estimates given by the proposed contact information model. The section concludes by presenting simulation results regarding the impact of contact information size on message forwarding, particularly in terms of successful contact information exchanges and energy consumption.

The second part (i.e., Section 5.2.2) evaluates the performance of PROPHET, PoiFord and HERO in terms of the selected metrics, namely throughput, average delivery delay, message transmission overhead, and average energy consumption. Specifically, PoiFord's ability to improve network performance while reducing contact information overhead is measured by comparing its results with those of PROPHET. The failure to implement the process of exchanging summary vectors in simulation experiments and its impact on the validity of performance evaluation is also discussed.

### 5.2.1. Contact Information Overhead

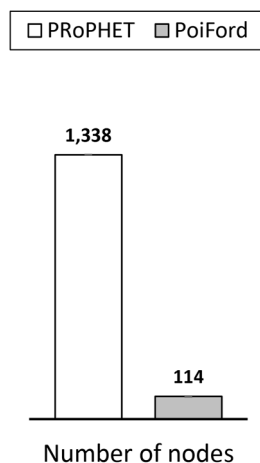
#### A. The Amount of Contact Information Maintained by PROPHET and PoiFord

The amount of contact information nodes exchanged per encounter and per day is shown in Figure 5. The results show that PoiFord is able to reduce contact information overhead when incorporated into an existing routing protocol. As shown in the figure, PROPHET requires an average of 48 KB contact information to be exchanged (i.e., sent plus received) during each encounter before messages can be forwarded. PoiFord reduces the average and maximum contact information size maintained on a PROPHET node by 77% and 54%, respectively. The figure also shows that, during an observation period of 1 day, the total amount of contact information exchanged by nodes running on PROPHET reaches an average and maximum of 15.6 MB and 97 MB, respectively. PoiFord reduces these values by 78% and 57%, respectively.



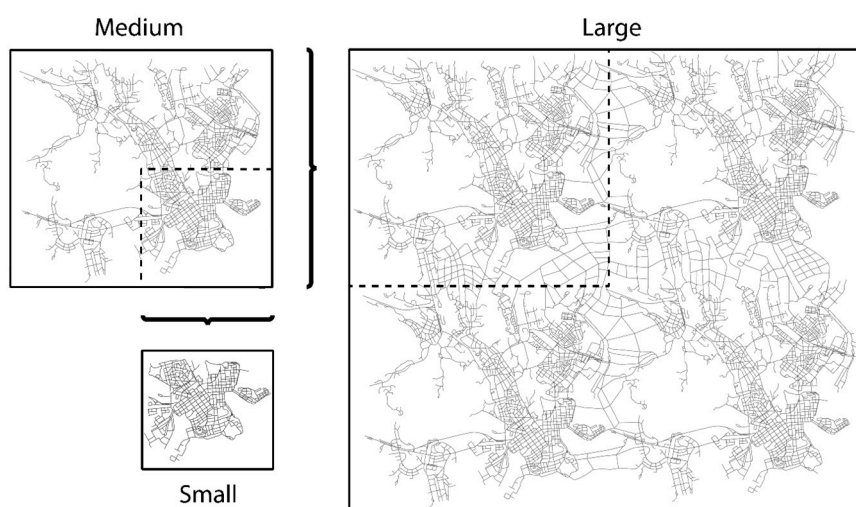
**Figure 5.** The size of contact information that nodes exchanged when running on PROPHET and PoiFord.

PoiFord is able to reduce contact information overhead due to the fact that it maintains encounter history of much fewer nodes—only about 9% of the number of nodes for which PROPHET needs to keep history of (cf. Figure 6). In addition to this, it is also important to know how contact information size responds to increasing network size and node population. In order to investigate this, we analyze PROPHET and PoiFord over three different movement scenarios of increasing network size in the next section.



**Figure 6.** The average number of nodes for which encounter history is maintained.

The (default) Helsinki simulation area in ONE simulator, which consists of four main districts, is modified to obtain three movement scenarios, namely small, medium, and large, correspond to their relative size. As shown in Figure 7, the Helsinki simulation area is used to represent the medium scenario, the small scenario is represented by one of the four main districts in the medium scenario, and the large scenario is artificially generated from the combination of four copies of the medium scenario. Nineteen, 76 and 304 mobile nodes move according to WDM in the small, medium, and large scenarios, respectively—the number of nodes is varied proportionally leaving the essential encounter characteristics unchanged. Next, we present the amount of contact information maintained by PROPHET and PoiFord over each scenario.

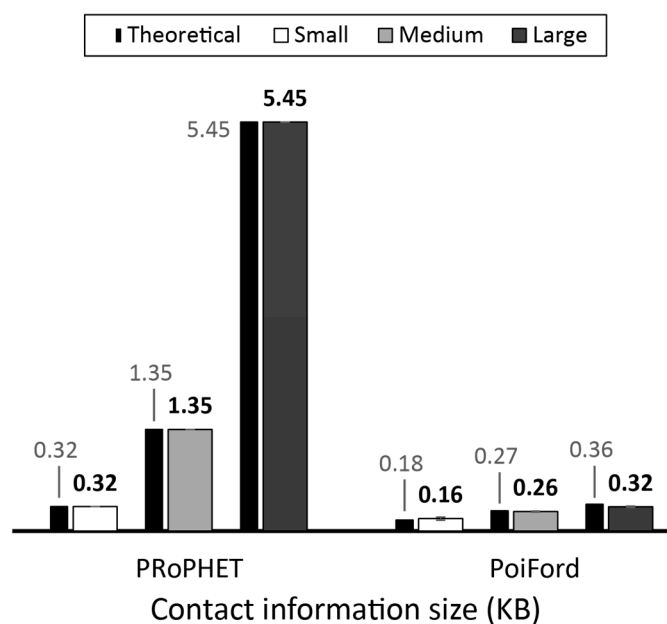


**Figure 7.** Movement scenarios (namely small, medium and large) used for evaluating the impact of increasing network size on the amount of contact information maintained by PROPHET and PoiFord.



## B. Impact of Increasing Network Size on The Amount of Contact Information Maintained by PProPHET and PoiFord

The simulation results presented here augment the contact information overhead models presented in Sections 3.1 and 5.1.3 for PProPHET and PoiFord, respectively. The following variables are used for computing contact information size with the models: (i)  $X_D$ , a 64-bit double; and (ii)  $X_S$ , a string of 80 bits (i.e., 5 characters). We consider 24 time slots for PoiFord, and take the number of POIs in the small, medium and large scenarios as 5, 12 and 35, respectively. Figure 8 shows the theoretical and simulation results for contact information size over increasing network size. The theoretical results obtained with the models agree with the simulation results.



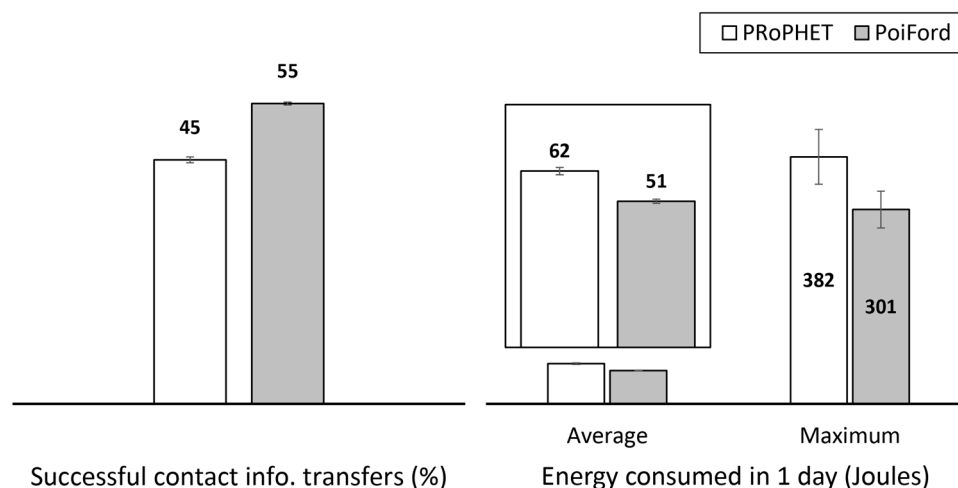
**Figure 8.** Theoretical and simulated average contact information size on nodes running on PProPHET and PoiFord over the small, medium and large scenario.

Owing to PProPHET's transitive property, contact information overhead rises steadily with the number of nodes in the network. Due to location-based influence, the contact information overhead incurred by PoiFord remains quite stable over different node populations. PoiFord's stability in this regard is due to the fixed size of its location table. The slight rise over higher node populations results from increased node density in POIs, requiring encounter history to be maintained for more nodes with mutual POIs. This rise becomes less noticeable as the number of POIs increases, which is likely in city-wide scenarios.

## C. Impact of Contact Information Size on Message Forwarding

Section 3.2 explained the impact of contact information size on message forwarding in terms of data exchange success and energy consumption on nodes. In order to verify these claims, we allow the simulation to run without generating data messages, so that only contact information is exchanged when nodes encounter each other. Hence, in the simulation, nodes consume energy due to exchanging contact information alone. In order to investigate the impact of contact information size on data exchange success, we record the percent of contact information transfer attempts that were successful when nodes run under PProPHET and PoiFord in Figure 9. Since nodes can forward messages only after a successful transfer of contact information, successfully transferring more number of contact information means more chances of exchanging messages. In other words, nodes running on PoiFord are able to forward more messages during encounters due to less contact information size.

Figure 9 also shows that this allows PoiFord to consume less energy per contact information transfer, as it reduces the average and maximum energy consumed with P<sub>Ro</sub>PHET in a day by 17.7% and 21.2%, respectively.



**Figure 9.** Percentage of contact information transfer attempts that were successful and energy consumed for transferring contact information in a day.

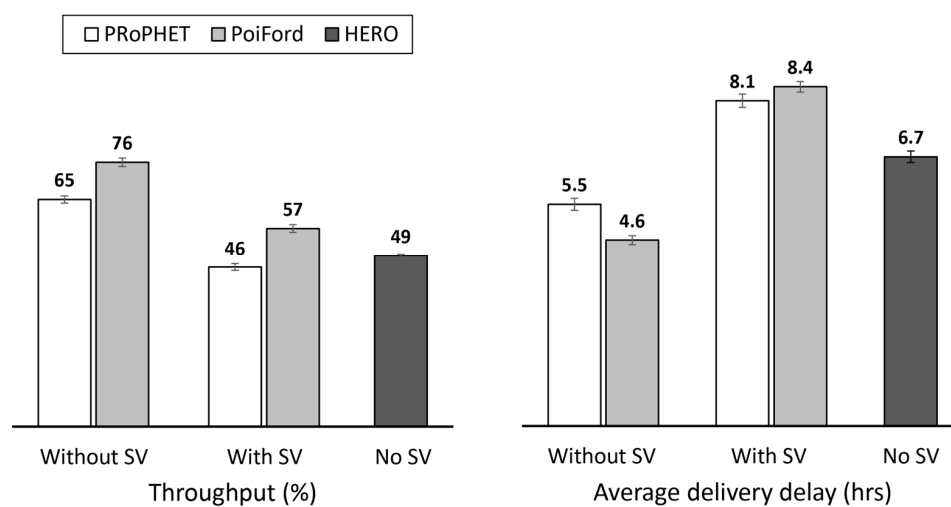
### 5.2.2. Routing Performance of PoiFord

With less contact information maintained on nodes, PoiFord is able to improve overall network performance significantly. To investigate how PoiFord improves routing performance, we run two versions of the simulation for P<sub>Ro</sub>PHET and PoiFord. The first version of the simulation namely “without SV” did not implement summary vectors, so that whenever two nodes encounter each other, they obtain the necessary encounter-based knowledge directly from the simulator’s internal memory. This is how the current version of P<sub>Ro</sub>PHET and most encounter-based routing protocols are implemented in ONE simulator—they abstract the process of exchanging summary vectors so a node can access the routing information stored in another node without receiving or sending summary vectors. For the second version of the simulation, namely “with SV”, we implement the process of exchanging summary vectors, so that whenever two nodes encounter each other, they actually generate and exchange new messages to which they attach a message vector and a contact vector (as stated earlier in Section 2.1). Hence, message forwarding takes place only after a summary vector has been successfully received. The results are recorded in Figure 10. “No SV” indicates that HERO does not require summary vectors to be exchanged between encountered nodes.

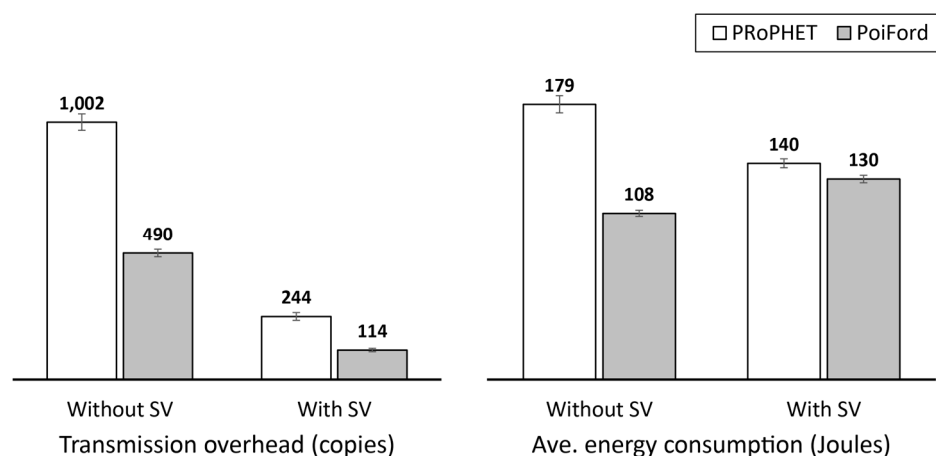
Without implementing summary vectors, incorporating PoiFord into P<sub>Ro</sub>PHET increases throughput by about 16.9% and reduces average delivery delay by about 16.4%. PoiFord is aware of a major and often overlooked factor, which is the adverse effects of spatial locality. Spatial locality increases with the geographical area of the network in the sense that most nodes travel relatively short trips and have high preference to a local area. This reduces the encounter opportunities between sets of nodes that reside in different geographical areas. In that case, finding suitable relays with forwarding utilities based on only direct encounter information may be challenging. Unfortunately, as mentioned earlier, the solution provided by transitive properties (such as P<sub>Ro</sub>PHET’s) may result in excess transmissions. With locality awareness from utilizing location-based knowledge, PoiFord is able to find more suitable relay nodes even in high degrees of spatial locality, thereby delivering more messages in less time. HERO, on the other hand, records the least throughput and highest average delivery delay due to the absence of any encounter-based or social-based knowledge.

With the implementation of summary vectors, PoiFord further increases throughput by 23.9%. This indicates that in addition to locality awareness, PoiFord’s ability to reduce contact information

overhead also contributes to increased throughput when summary vectors are considered—which is a more realistic case. With smaller summary vectors, nodes running on PoiFord are able to forward more messages during each encounter opportunity. As a result, less messages are lost and throughput is increased at the cost of slightly (3.7%) higher average delivery delay, since the metric accounts for only delivered messages. It can also be observed from Figure 10 that HERO records a higher throughput than PProPHET in this scenario. This shows that results obtained without actually implementing summary vector exchange could be exaggerating—this is also observed in Figure 11. This practice may provide misleading results especially when the performance of different routing protocols is being compared. Neglecting summary vector exchange does not only affect the comparison with location-based forwarding techniques (e.g., comparing the performance of PProPHET and HERO), but also the comparison with other types of forwarding approaches (e.g., PProPHET and PoiFord). Hence, it would be a better practice to implement summary vector exchange when evaluating the performance of OppNet protocols via simulations.



**Figure 10.** Throughput and average delivery delay of PProPHET, PoiFord and HERO.

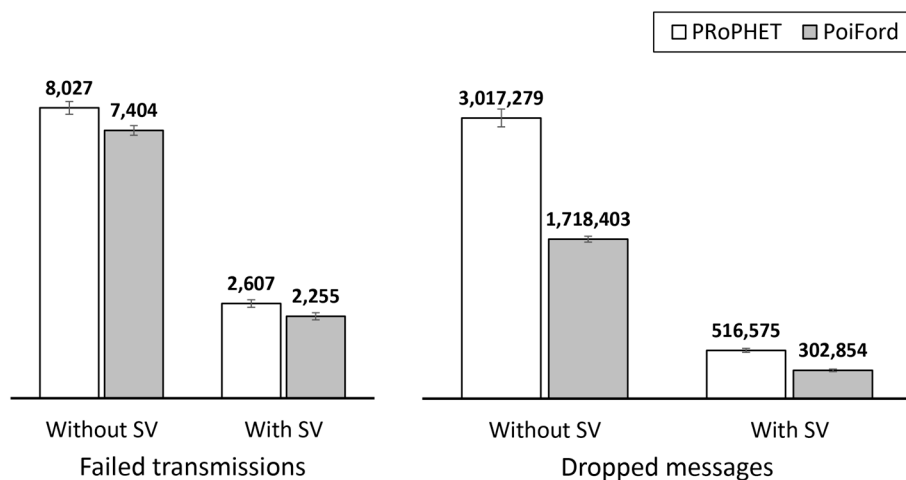


**Figure 11.** Message transmission overhead and average energy consumption of PProPHET and PoiFord (note that energy consumption in these results is mainly for purpose of comparison, and does not represent realistic energy values).

There are also other means through which PoiFord improves the performance of a routing protocol it is incorporated into. Locality awareness allows messages to be forwarded to nodes moving

closer to the destination's POI. With delivery predictability alone, the transitive property allows more nodes to present higher forwarding utilities, even by slight margins. Hence, nodes receive more messages and message transmission overhead is increased (cf. Figure 11). PoiFord reduces message transmission overhead by about 51% and 53% without and with the implementation of summary vectors, respectively. PoiFord also reduces energy consumption by about 40% and 7% likewise—note that PoiFord's energy reduction tends to increase with increasing network size due to less contact information and fewer message transmissions. It can also be observed from Figure 11 that PProPHET's message transmission overhead and energy consumption is less when summary vector exchange is implemented. This is because fewer messages are successfully forwarded when summary vectors need to be exchanged beforehand. This is also the case for PoiFord's message transmission overhead. In terms of energy, however, PoiFord records more consumption when summary vector exchange is implemented. This is due to the additional energy consumed in successfully exchanging more number of summary vectors. In the case of PProPHET, the message transmission overhead when summary vector exchange is not implemented is so high that the energy consumed in the process exceeds this additional energy consumption.

PProPHET's high message transmission overhead due to utilizing the encounter-based delivery predictability alone leads to more number of failed transmissions (cf. Figure 12), as nodes carry more messages and try to forward all of them within limited encounter durations. The resulting increase in transmissions not only expends more energy on nodes (cf. Figure 11), but also leads to increased contention that degrades throughput. Using delivery predictability alone may also degrade throughput since messages are more likely to be dropped due to increased buffer occupancy and overflows. The combination of encounter and location-based knowledge allows PoiFord's forwarding utility to identify a wider range of more suitable relay nodes, thereby reducing failed message transmissions and message drops (cf. Figure 12).



**Figure 12.** Number of failed transmissions (aka. aborted messages in ONE simulator) and number of dropped messages for PProPHET and PoiFord.

## 6. Conclusions and Future Work

Due to the distributed nature of OppNets, most routing protocols guarantee acceptable throughput by following an approach that requires contact information to be maintained for every encountered node and exchanged via summary vectors during each encounter. However, this exchange of contact information is often neglected in OppNet simulations. For instance, PProPHET routing protocol in the latest release of ONE simulator (v1.6.0) does not implement summary vector exchange when nodes encounter each other. Nodes rather obtain the necessary encounter-based knowledge directly from the simulator's internal memory, which is globally and readily available. Unfortunately, this

unrealistic abstraction has prevented researchers from gaining insight into the impact of contact information exchange on routing performance. Since opportunistic IoT scenarios envision a wide range of participating nodes (e.g., portable handheld devices, sensors, smart vehicles, appliances and other infrastructure), high node population and encounter frequency may result in large amounts of contact information and frequent exchanges, respectively.

Our contribution in this regard is twofold:

1. First, we modelled contact information overhead and investigated its impact on routing performance. We observed that the size of contact information exchanged when nodes encounter each other is related to routing performance in terms of data exchange success as well as the energy utilized in the process. We verified these observations via experiments in ONE simulator, which indicated that exchanging more contact information reduces the number of data messages that can be forwarded per encounter. The results also indicated that exchanging more contact information consumes more energy, especially in high node population and encounter frequency.
2. Second, we proposed PoiFord, a message forwarding algorithm that can be incorporated into existing encounter-based routing protocols to reduce contact information overhead without compromising throughput. The operation of PoiFord is based on the spatial and temporal regularity embedded in human movement and its design is fully distributed and does not require global knowledge or central management. PoiFord reduces contact information overhead by maintaining encounter history for only nodes that have at least one mutual POI. Suitable forwarders are selected based on a forwarding utility derived from the combination of location-based information and available encounter-based information. Incorporating PoiFord into PRoPHET improved routing performance in terms of throughput, message transmission overhead, and energy consumption.

Lessons learned from this study include:

- Contact information may account for a significant portion of summary vectors, especially in scenarios with high node population;
- The energy utilized in exchanging contact information increases with node population and encounter frequency;
- Neglecting the exchange of summary vectors in simulations may lead to exaggerated results and unfair performance comparison—a protocol that performed better may perform worse when summary vector exchange is implemented;
- Maintaining history of every encounter may degrade routing performance and is not required to achieve an acceptable overall performance; and
- A suitable forwarding utility can be derived without using the transitive property—since the transitive property requires a history of every encounter to be maintained.

We expect our results and findings to motivate further research contributions. Since this work incorporated PoiFord into PRoPHET, future work could investigate the contact information overhead reduction, as well as overall network performance improvement for different encounter-based routing protocols. Investigating the impact of using different values of  $r$  on different sets of nodes running on PoiFord is also reserved for future work, as this will represent a more realistic configuration such as in real-world implementation.

Future work could also address the energy consumed in taking GPS readings. Due to energy consumption, users may have the GPS on their device turned off and may not be willing to turn it back on just to answer the needs of a specific OppNet routing solution. Since human movement exhibits a reasonable amount of regularity, it may not be required to take GPS readings regularly. With mechanisms for learning user movement patterns, locations for a number of slots in the location table could be predicted. However, some users may not present much regularity in their movement (e.g., salesmen), hence, may require alternative solutions.



In real-world implementation, requiring nodes to exchange user POI information for determining encounter relevance (i.e., to identify neighbors for which to maintain and update encounter history) may raise privacy concerns. Future contributions can be made by proposing alternative means of determining relevant encounters. Real-world implementation also raises the issue of unreliable GPS signal in geographically restricted locations. Although we provided a workaround by approximating the current location as the previous record in the location table, GPS signal is assumed to be always available in our experiments. Investigating the impact of this solution has been left for future work due to current limitations of the simulation tool.

**Acknowledgments:** This research was supported by Ministry of Education (MOE) Malaysia and conducted in collaboration with Research Management Center (RMC) at Universiti Teknologi Malaysia (UTM) under VOT NUMBER: R.J130000.7828.4F784.

**Author Contributions:** Tekenate E. Amah conceived the original idea, designed the figures, implemented the proposals in the simulator and wrote the manuscript with input from all authors. Antonio Oliveira Jr. encouraged Tekenate E. Amah to investigate energy consumption and verified the related model. Waldir Moreira contributed to the problem background the evaluation methodology. Aliyu M. Abali assisted with the simulation set up and helped carry out the experiments. Syed Othmawi Abd Rahman and Muhammad Hafiz Mohammed were involved in processing the experimental data. Maznah Kamat and Kamalrulnizam Abu Bakar aided in interpreting the results and supervised the findings of this work. All authors provided critical feedback and helped shape the research, analysis and manuscript.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Heinemann, A.; Straub, T. Opportunistic Networks as an Enabling Technology for Mobile Word-of-Mouth Advertising. In *Handbook of Research on Mobile Marketing Management*; Pousttchi, K., Wiedemann, D., Eds.; Business Science Reference: Hershey, PA, USA, 2010; pp. 236–254.
2. Zhang, B.; Teng, J.; Bai, X.; Yang, Z.; Xuan, D. P3-coupon: A Probabilistic System for Prompt and Privacy-Preserving Electronic Coupon Distribution. In Proceedings of the IEEE International Conference on Pervasive Computing and Communications, Seattle, WA, USA, 21–25 March 2011; pp. 93–101.
3. Guo, B.; Yu, Z.; Zhou, X.; Zhang, D. Opportunistic IoT: Exploring the social side of the internet of things. In Proceedings of the 16th IEEE International Conference on Computer Supported Cooperative Work in Design, Wuhan, China, 23–25 May 2012; pp. 925–929.
4. Santi, P. Next Generation Wireless Networks. In *Mobility Models for Next Generation Wireless Networks: Ad Hoc, Vehicular and Mesh Networks*, 1st ed.; Hutchison, D., Fdida, S., Sventek, J., Eds.; John Wiley & Sons, Ltd.: Chichester, UK, 2012; pp. 1–17.
5. Das Deb, D.; Bose, S.; Bandyopadhyay, S. Coordinating Disaster Relief Operations Using Smart Phone/PDA Based Peer-To-Peer Communication. *Int. J. Wirel. Mob. Netw.* **2012**, *4*, 27–44. [[CrossRef](#)]
6. Krug, S.; Begerow, P.; Al Rubaye, A.; Schellenberg, S.; Seitz, J. A Realistic Underlay Concept for Delay Tolerant Networks in Disaster Scenarios. In Proceedings of the 10th International Conference on Mobile Ad-hoc and Sensor Networks, Maui, HI, USA, 19–21 December 2014; pp. 163–170.
7. ITU-T Focus Group on Smart Sustainable Cities. An Overview of Smart Sustainable Cities and the Role of Information and Communication Technologies. Available online: [http://www.itu.int/en/ITU-T/focusgroups/ssc/Documents/Approved\\_Deliverables/TR-Overview-SSC.docx](http://www.itu.int/en/ITU-T/focusgroups/ssc/Documents/Approved_Deliverables/TR-Overview-SSC.docx) (accessed on 2 September 2017).
8. Gomez, C.; Paradells, J. Urban Automation Networks: Current and Emerging Solutions for Sensed Data Collection and Actuation in Smart Cities. *Sensors* **2015**, *15*, 22874–22898. [[CrossRef](#)] [[PubMed](#)]
9. Clarke, R. Smart Cities and the Internet of Everything: The Foundation for Delivering Next-Generation Citizen Services. Available online: [http://www.cisco.com/c/dam/en\\_us/solutions/industries/docs/scc/ioe\\_citizen\\_svcs\\_white\\_paper\\_idc\\_2013.pdf%20](http://www.cisco.com/c/dam/en_us/solutions/industries/docs/scc/ioe_citizen_svcs_white_paper_idc_2013.pdf%20) (accessed on 2 September 2017).
10. Du, W.; Li, Z.; Liando, J.; Li, M. From Rateless to Distanceless: Enabling Sparse Sensor Network Deployment in Large Areas. *IEEE/ACM Trans. Netw.* **2016**, *24*, 2498–2511. [[CrossRef](#)]
11. Mao, X.; Miao, X.; He, Y.; Li, X.; Liu, Y. CitySee: Urban CO2 monitoring with sensors. In Proceedings of the 31st Annual IEEE International Conference on Computer Communications, Orlando, FL, USA, 25–30 March 2012; pp. 1611–1619.

12. Valerio, L.; Bruno, R.; Passarella, A. Cellular traffic offloading via opportunistic networking with reinforcement learning. *Comput. Commun.* **2015**, *71*, 129–141. [[CrossRef](#)]
13. Ochiai, H.; Ishizuka, H.; Kawakami, Y.; Esaki, H. Agent based sensor data gathering for agricultural applications. *IEEE Sens. J.* **2011**, *11*, 2861–2868. [[CrossRef](#)]
14. Tovar, A.; Friesen, T.; Ferens, K.; McLeod, B. A DTN wireless sensor network for wildlife habitat monitoring. In Proceedings of the 23rd Canadian Conference on Electrical and Computer Engineering, Calgary, AB, Canada, 2–5 May 2010; pp. 1–5.
15. McDonald, P.; Geraghty, D.; Humphreys, I.; Farrell, S.; Cahill, V. Sensor Network with Delay Tolerance (SeNDT). In Proceedings of the 16th International Conference on Computer Communications and Networks, Honolulu, HI, USA, 13–16 August 2007; pp. 1333–1338.
16. Velásquez-Villada, C.; Donoso, Y. Delay/Disruption Tolerant Network-Based Message Forwarding for a River Pollution Monitoring Wireless Sensor Network Application. *Sensors* **2016**, *16*, 436. [[CrossRef](#)] [[PubMed](#)]
17. Cheng, N.; Lu, N.; Zhang, N.; Shen, X.; Mark, J. Vehicle-assisted data delivery for smart grid: An optimal stopping approach. In Proceedings of the IEEE International Conference on Communications, Budapest, Hungary, 9–13 June 2013; pp. 6184–6188.
18. Park, U.; Heidemann, J. Data muling with mobile phones for sensor networks. In Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems, Seattle, WA, USA, 1–4 November 2011; pp. 162–175.
19. Can, Z.; Demirbas, M. Smartphone-based data collection from wireless sensor networks in an urban environment. *J. Netw. Comput. Appl.* **2015**, *58*, 208–216. [[CrossRef](#)]
20. Shi, F.; Adeel, U.; Theodoridis, E.; Haghighi, M.; McCann, J. OppNet: Enabling citizen centric urban IoT data collection through opportunistic connectivity service. In Proceedings of the IEEE 3rd World Forum on Internet of Things, Reston, VA, USA, 12–14 December 2016; pp. 723–728.
21. Dimatteo, S.; Hui, P.; Han, B.; Li, V. Cellular Traffic Offloading through Wi-Fi Networks. In Proceedings of the IEEE 8th International Conference on Mobile Ad-Hoc and Sensor Systems, Valencia, Spain, 17–22 October 2011; pp. 192–201.
22. Petz, A.; Lindgren, A.; Hui, P.; Julien, C. Madserver: A server architecture for mobile advanced delivery. In Proceedings of the 7th ACM International Workshop on Challenged Networks, Istanbul, Turkey, 22–26 August 2012; pp. 17–22.
23. Rebecchi, F.; Dias de Amorim, M.; Conan, V.; Passarella, A.; Bruno, R.; Conti, M. Data Offloading Techniques in Cellular Networks: A Survey. *IEEE Commun. Surv. Tutor.* **2015**, *17*, 580–603. [[CrossRef](#)]
24. Aguilar, S.; Vidal, R.; Gomez, C. Opportunistic Sensor Data Collection with Bluetooth Low Energy. *Sensors* **2017**, *17*, 159. [[CrossRef](#)] [[PubMed](#)]
25. Wu, X.; Brown, K.; Sreenan, C. Analysis of smartphone user mobility traces for opportunistic data collection in wireless sensor networks. *Pervasive Mob. Comput.* **2013**, *9*, 881–891. [[CrossRef](#)]
26. Wu, X.; Brown, K.; Sreenan, C. Exploiting Rush Hours for Energy-Efficient Contact Probing in Opportunistic Data Collection. In Proceedings of the 31st International Conference on Distributed Computing Systems Workshops, Minneapolis, MN, USA, 20–24 June 2011; pp. 240–247.
27. Mashhadi, A.; Mokhtar, S.; Capra, L. Fair content dissemination in participatory DTNs. *Ad Hoc Netw.* **2012**, *10*, 1633–1645. [[CrossRef](#)]
28. Mtibaa, A.; Harras, K. Fairness-related challenges in mobile opportunistic networking. *Comput. Netw.* **2013**, *57*, 228–242. [[CrossRef](#)]
29. Wang, S.; Liu, M.; Cheng, X.; Li, Z.; Huang, J.; Chen, B. HERO—A Home Based Routing in Pocket Switched Networks. In Proceedings of the International Conference on Wireless Algorithms, Systems, and Applications, Yellow Mountains, China, 8–10 August 2012; pp. 20–30.
30. Anastasiades, C.; Barun, T.; Siris, V. Information-Centric Networking in Mobile and Opportunistic Networks. In *Wireless Networking for Moving Objects: Protocols, Architectures, Tools, Services and Applications*; Springer: Cham, Switzerland, 2014; pp. 14–30.
31. Ott, J.; Hyytiä, E.; Lassila, P.; Kangasharju, J.; Santra, S. Floating content for probabilistic information sharing. *Pervasive Mob. Comput.* **2011**, *7*, 671–689. [[CrossRef](#)]
32. Foerster, A.; Udugama, A.; Görg, C.; Kuladinithi, K.; Timm-Giel, A.; Cama-Pinto, A. A Novel Data Dissemination Model for Organic Data Flows. In Proceedings of the 7th International Conference on Mobile Networks and Management, Santander, Spain, 16–18 September 2015; pp. 239–252.

33. Vahdat, A.; Becker, D. Epidemic Routing for Partially Connected Ad Hoc Networks. Available online: <http://issg.cs.duke.edu/epidemic/epidemic.pdf> (accessed on 2 September 2017).
34. Leguay, J.; Friedman, T.; Conan, V. Evaluating Mobility Pattern Space Routing for DTNS. In Proceedings of the 25th IEEE International Conference on Computer Communications, Barcelona, Spain, 23–29 April 2006; pp. 1–10.
35. Mtibaa, A.; May, M.; Diot, C.; Ammar, M. Peoplerank: Social opportunistic forwarding. In Proceedings of the IEEE INFOCOM, San Diego, CA, USA, 14–19 March 2010; pp. 1–5.
36. Hui, P.; Crowcroft, J.; Yoneki, E. Bubble rap: Social-based forwarding in delay-tolerant networks. *IEEE Trans. Mob. Comput.* **2011**, *10*, 1576–1589. [[CrossRef](#)]
37. Moreira, W.; Mendes, P.; Sargento, S. Opportunistic routing based on daily routines. In Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, San Francisco, CA, USA, 25–28 June 2012; pp. 1–6.
38. Nguyen, H.; Giordano, S. Context information prediction for social-based routing in opportunistic networks. *Ad Hoc Netw.* **2012**, *10*, 1557–1569. [[CrossRef](#)]
39. Grasic, S.; Davies, E.; Lindgren, A.; Doria, A. The evolution of a DTN routing protocol—Prophetv2. In Proceedings of the 6th ACM workshop on challenged networks, Las Vegas, NV, USA, 19–23 September 2011; pp. 27–30.
40. Wang, X.; He, R.; Lin, B.; Wang, Y. Probabilistic Routing Based on Two-Hop Information in Delay/Disruption Tolerant Networks. *J. Electr. Comput. Eng.* **2015**. [[CrossRef](#)]
41. Sadat, N.; Tasnim, M. A Neighborhood Contact History Based Spraying Heuristic for Delay Tolerant Networks. In Proceedings of the IEEE 3rd International Conference on Informatics, Electronics and Vision, Dhaka, Bangladesh, 23–24 May 2014; pp. 1–5.
42. Spyropoulos, T.; Psounis, K.; Raghavendra, C. Spray and Focus: Efficient Mobility-Assisted Routing for Heterogeneous and Correlated Mobility. In Proceedings of the 5th Annual IEEE International Conference on Pervasive Computing and Communications Workshops, White Plains, NY, USA, 19–23 March 2007; pp. 79–85.
43. Rolla, V.; Curado, M. Time Message System for Delay Tolerant Networks. In Proceedings of the 2nd Baltic Congress on Future Internet Communications, Vilnius, Lithuania, 25–27 April 2012; pp. 239–245.
44. Liu, C.; Wu, J. An optimal probabilistic forwarding protocol in delay tolerant networks. In Proceedings of the 10th ACM International Symposium on Mobile Ad Hoc Networking and Computing, New Orleans, LA, USA, 18–21 May 2009; pp. 105–114.
45. Lindgren, A.; Doria, A. Experiences from Deploying a Real-Life DTN System. In Proceedings of the 4th IEEE Consumer Communications and Networking Conference, Las Vegas, NV, USA, 11–13 January 2007; pp. 217–221.
46. Davies, E. Delay-Tolerant Networking: Moving towards Real-World Deployment. Available online: [http://www.folly.org.uk/Papers/dtn\\_slovenia\\_paper\\_10.pdf](http://www.folly.org.uk/Papers/dtn_slovenia_paper_10.pdf) (accessed on 2 September 2017).
47. Amah, T.; Kamat, M.; Bakar, K.; Moreira, W.; Oliveira, A.; Batista, M. Spatial Locality in Pocket Switched Networks. In Proceedings of the IEEE 17th International Symposium on a World of Wireless, Mobile and Multimedia Networks, Coimbra, Portugal, 21–24 June 2016; pp. 1–6.
48. Yu, C.; Bao, C.; Jin, H. Hierarchical geographical tags based routing scheme in delay/disruption tolerant mobile ad hoc networks. In Proceedings of the International Conference on Human Centered Computing, Phnom Penh, Cambodia, 27–29 November 2014; pp. 352–364.
49. Wang, S.; Liu, M.; Cheng, X.; Li, Z.; Huang, J.; Chen, B. Opportunistic routing in intermittently connected mobile P2P networks. *IEEE J. Sel. Areas Commun.* **2013**, *31*, 369–378. [[CrossRef](#)]
50. Balazinska, M.; Castro, P. Characterizing mobility and network usage in a corporate wireless local-area network. In Proceedings of the 1st ACM International Conference on Mobile Systems, Applications and Services, San Francisco, CA, USA, 5–8 May 2003; pp. 303–316.
51. Henderson, T.; Kotz, D.; Abyzov, I. The changing usage of a mature campus-wide wireless network. *Comput. Netw.* **2008**, *52*, 2690–2712. [[CrossRef](#)]
52. Tian, Y.; Li, J. Location-aware routing for delay tolerant networks. In Proceedings of the 5th International ICST Conference on Communications and Networking, Beijing, China, 25–27 August 2010; pp. 1–5.

53. Khan, S.; Loo, J.; Lasebae, A.; Azam, M.; Adeel, M.; Kausar, R.; Sardar, H. LOC algorithm: Location-aware opportunistic forwarding by using node's approximate location. *Int. J. Pervasive Comput. Commun.* **2014**, *10*, 481–496. [CrossRef]
54. Amah, T.; Kamat, M.; Moreira, W.; Abu Bakar, K.; Mandala, S.; Batista, M. Towards next-generation routing protocols for pocket switched networks. *J. Netw. Comput. Appl.* **2016**, *70*, 51–88. [CrossRef]
55. About Us—Ecotourism Campus, Universiti Teknologi Malaysia. Available online: <http://www.utm.my/ecotourism/about-us> (accessed on 2 September 2017).
56. Facts and Figures—About UTM, Universiti Teknologi Malaysia. Available online: <http://www.utm.my/about/facts-and-figures> (accessed on 2 September 2017).
57. Smith, P. Comparing Low-Power Wireless Technologies. Available online: <https://www.digikey.com/en/articles/techzone/2011/aug/comparing-low-power-wireless-technologies> (accessed on 2 September 2017).
58. Lindgren, A.; Doria, A.; Davies, E.; Grasic, S. Probabilistic Routing Protocol for Intermittently Connected Networks. Available online: <http://www.rfc-editor.org/info/rfc6693> (accessed on 2 September 2017).
59. Cho, E.; Myers, S.; Leskovec, J. Friendship and mobility: User movement in location based social networks. In Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Diego, CA, USA, 21–24 August 2011; pp. 1082–1090.
60. Keränen, A.; Ott, J.; Kärkkäinen, T. The ONE simulator for DTN protocol evaluation. In Proceedings of the 2nd International Conference on Simulation Tools and Techniques, Rome, Italy, 2–6 March 2009; p. 55.
61. Ekman, F.; Keränen, A.; Karvo, J.; Ott, J. Working day movement model. In Proceedings of the 1st ACM SIGMOBILE Workshop on Mobility Models, Hong Kong, China, 26–30 May 2008; pp. 33–40.
62. Silva, D.; Costa, A.; Macedo, J. Energy impact analysis on dtn routing protocols. In Proceedings of the 4th Extreme Conference on Communication, Zürich, Switzerland, 10–14 March 2012.



© 2017 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).